

---

Subject: Re: GridCtrl performance

Posted by [Sender Ghost](#) on Thu, 25 Apr 2013 14:47:23 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

Hello.

crydev wrote on Mon, 22 April 2013 22:25BioBytes wrote on Mon, 22 April 2013 21:43Hello  
Crydev,

Did you try SetVirtualCount method in ArrayCtrl ?

Regards

Biobytess

Yes I tried using it but unfortunately it didn't solve my problem. If I could safely use the multithreaded structure to fill up my GridCtrl it would speed up very much but because it is GUI it isn't possible.

I don't know how you tried it, but try again.

Below is additional example of VirtualArray reference application:

Toggle Spoiler

```
#include <CtrlLib/CtrlLib.h>

using namespace Upp;

int count = 0x100000;
Vector<String> data;

template <String (GetData) (int index)>
struct NumberToText : public Convert {
    virtual Value Format(const Value& q) const {
        return GetData(int(q));
    }
};

String GetIndexData(int index)
{
    ASSERT(index >= 0 && index < data.GetCount());
    return data[index];
}

String GetReverseData(int index)
{
    ASSERT(index >= 0 && index < data.GetCount());
    return data[count - index - 1];
}
```

```

String FormatData(int index)
{
    return NFormat("Data #%" + index + 1);
}

class App : public TopWindow {
public:
    typedef App CLASSNAME;
    App();

    // Ctrl
    ArrayCtrl list;
    Button btnAdd, btnRemove;
    // Events
    void OnAdd();
    void OnRemove();
};

const int addition = count / 2;

void App::OnAdd()
{
    const int prevCount = count;
    count += addition;
    data.SetCount(count);

    for (int i = prevCount; i < count; ++i)
        data[i] = FormatData(i);

    list.SetVirtualCount(count);
}

void App::OnRemove()
{
    count -= addition;
    if (count < 0)
        count = 0;

    data.SetCount(count);
    list.SetVirtualCount(count);
}

App::App()
{
    Title("Virtual ArrayCtrl");
    Sizeable().Zoomable();
    const Size sz(640, 480);
    SetRect(sz); SetMinSize(sz);
}

```

```

btnAdd.SetLabel("Add") <=> THISBACK(OnAdd);
btnRemove.SetLabel("Remove") <=> THISBACK(OnRemove);

HeaderCtrl::Column& hc = list.AddRowNumColumn("Index").HeaderTab();
const int textWidth = GetTextSize(AsString(count), Draw::GetStdFont()).cx,
columnWidth = textWidth + hc.GetMargin() * 2 + 1;
hc.Fixed(columnWidth);

list.AddRowNumColumn("Data", 50).SetConvert(Single<NumberToText<GetIndexData>>());
list.AddRowNumColumn("Reverse Data",
50).SetConvert(Single<NumberToText<GetReverseData>>());
list.SetVirtualCount(count);

const int offset = 75;
Add(btnAdd.TopPosZ(4, 20).LeftPosZ(4, offset));
Add(btnRemove.TopPosZ(4, 20).LeftPosZ(offset + 8, offset));
Add(list.HSizePosZ(4, 4).VSizePosZ(28, 4));
}

GUI_APP_MAIN
{
Ctrl::GlobalBackPaint();

data.SetCount(count);
{
Progress p;
p.SetText("Preparing the data..");
for (int i = 0; i < count; ++i) {
data[i] = FormatData(i);
p.Set(i, count);
}
}
}

App app;
app.Run();
}

```

Where instead of `Vector<String>` you could use your "faster" container, filled by parallel, if you want. Or even try to show real time (or cached) results, calculated by some index, without using any container.

## File Attachments

- 
- 1) [VirtualArrayCtrl.png](#), downloaded 645 times
-