Subject: Re: Know what you're using. Size of some common types.
Posted by Lance on Tue, 11 Jan 2022 21:07:07 GMT
View Forum Message <> Reply to Message

If ArrayCtrl limits children to Fake Ctrls only, SyncLineCtrls etc will be dispensible.

Back to Buttons, some chain of thoughts. Instead of inheriting from Ctrl, it should be leave as an interface (no data members); concrete Ctrls will decide whether to inherit from Ctrl, ArrayCtrl, EditField or other Ctrl derivatives and implementing the interface to use the Ctrl's Paint & other virtual functions to fake out Buttons.

This will add a cost of sizeof(void *) (pointer to interface vtbl) to each object of Classes that implement the interface. Why not add the interface (a set of virtual functions) to Ctrl directly to save this vtble pointer?

```
class Ctrl : public ...
{
public:
    virtual void Paint(Draw& w)
    {
        for(int i = 0; i < GetFakeCtrlsCount(); ++i)
        {
            Rect r=GetFakeCtrlsRect(i);
            w.Clipoff(r);
            PaintFakeCtrls(w, i);
            w.End();
        }
    }
    virtual void LeftDown(Point p, dword flag)
    {
        for(int i = 0; i < GetFakeCtrlsCount(); ++i)
        {
            Rect r=GetFakeCtrlsRect(i);
            if( MouseInRect(r) )
            {
                p.x -= r.left; p.y -= r.top;
                FakeCtrlsLeftDown(p, flag, i);
                break;
            }
        }
    }
    ...
     // added to support fake Ctrls
     virtual int GetFakeCtrlsCount()const{ return 0; }
     virtual void PaintFakeCtrls(Draw& w, int index){}
     virtual void FakeCtrlsLeftDown(Point p, dword flag, int index){}
     ... and some more virtual function of Ctrl with an extra parameter index.
};
```

This way we don't explicitly limit the type of fake Ctrls (Not even by a non-restrictive but suggestive name "Buttons"). It's totally up to each derivative that actually implements the interface to decide what Ctrls it wants to fake.

PS: instead of repeatedly calling multiple virtual functions to Paint each fake Ctrl, it make sense to instead

```
class Ctrl: ...
{
public:
    ...
      virtual void PaintFakeCtrls(Draw& w);
     ...
```

Then the PaintFakeCtrls() will be quite like ScollBar::Paint(). I don't know, let's see how you do it clearly and efficiently.