
Subject: virtual void Ctrl::PaintOver(Draw& w) new method

Posted by [tojocky](#) on Tue, 18 Jan 2011 14:15:04 GMT

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

Hello All,

First of all I wan to thank to all who contribute in CtrlCore developing. All the code is written very clear.

According by this question I propose to add new method "virtual void Ctrl::PaintOver(Draw& w)".

This method will call for painting after all child controls are already painted.
This method need to me for paint a highlight effect over all painted controls.

Now the source code:

Please change the method:

```
void Ctrl::CtrlPaint(SystemDraw& w, const Rect& clip) {
    GuiLock __;
    LEVELCHECK(w, this);
    LTIMING("CtrlPaint");
    Rect rect = GetRect().GetSize();
    Rect orect = rect.Inflated(overpaint);
    if(!IsShown() || orect.IsEmpty() || clip.IsEmpty() || !clip.Intersects(orect))
        return;
    Ctrl *q;
    Rect view = rect;
    for(int i = 0; i < frame.GetCount(); i++) {
        LEVELCHECK(w, NULL);
        frame[i].frame->FramePaint(w, view);
        view = frame[i].view;
    }
    Rect oview = view.Inflated(overpaint);
    bool hasviewctrls = false;
    bool viewexcluded = false;
    for(q = firstchild; q; q = q->next)
        if(q->IsShown())
            if(q->InFrame()) {
                if(!viewexcluded && IsTransparent() && q->GetRect().Intersects(view)) {
                    w.Begin();
                    w.ExcludeClip(view);
                    viewexcluded = true;
                }
                LEVELCHECK(w, q);
                Point off = q->GetRect().TopLeft();
                w.Offset(off);
                q->CtrlPaint(w, clip - off);
                w.End();
            }
}
```

```

else
    hasviewctrls = true;
if(viewexcluded)
    w.End();
DOLEVELCHECK;
if(!oview.IsEmpty()) {
    if(oview.Intersects(clip) && w.IsPainting(oview)) {
        LEVELCHECK(w, this);
        if(overpaint) {
            w.Clip(oview);
            w.Offset(view.left, view.top);
            Paint(w);
            PaintCaret(w);
            w.End();
            w.End();
        }
        else {
            w.Clipoff(view);
            Paint(w);
            PaintCaret(w);
            w.End();
        }
    }
}
if(hasviewctrls && !view.IsEmpty()) {
    Rect cl = clip & view;
    w.Clip(cl);
    for(q = firstchild; q; q = q->next)
        if(q->IsShown() && q->InView()) {
            LEVELCHECK(w, q);
            Rect qr = q->GetRect();
            Point off = qr.TopLeft() + view.TopLeft();
            Rect ocl = cl - off;
            if(ocl.Intersects(Rect(qr.GetSize()).Inflated(overpaint))) {
                w.Offset(off);
                q->CtrlPaint(w, cl - off);
                w.End();
            }
        }
    w.End();
}
w.Clipoff(rect);
PaintOver(w);
PaintCaret(w);
w.End();
}

```

And add new method in header:

```
class Ctrl : public Pte<Ctrl> {  
...  
    virtual void  PaintOver(Draw& w);  
...  
}
```

and source:

```
void Ctrl::PaintOver(Draw& draw)          {}
```

Maybe I'm wrong, I'm ready to discuss about this.

Thank you in advance!

Subject: Re: virtual void Ctrl::PaintOver(Draw& w) new method
Posted by [mirek](#) on Tue, 18 Jan 2011 17:27:40 GMT
[View Forum Message](#) <> [Reply to Message](#)

I am afraid such implementation would only work for transparent child widgets. Check PaintOpaqueAreas. (But I might be wrong).

So far, the 'canonical' method to achieve this is to use transparent overlay child at the end of list.

Mirek

Subject: Re: virtual void Ctrl::PaintOver(Draw& w) new method
Posted by [tojocky](#) on Tue, 18 Jan 2011 18:20:53 GMT
[View Forum Message](#) <> [Reply to Message](#)

mirek wrote on Tue, 18 January 2011 19:27I am afraid such implementation would only work for transparent child widgets. Check PaintOpaqueAreas. (But I might be wrong).

So far, the 'canonical' method to achieve this is to use transparent overlay child at the end of list.

Mirek

Maybe exist other method to achieve this?
