

---

Subject: Re: RectTracker filled with black  
Posted by [cbpporter](#) on Wed, 20 Feb 2008 08:33:57 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Since nobody commented on the coordinate relation between mouse cursor position/rect coordinates and the way the drawn rect visually "touches" the cursor (and also KeepRation(true) seems to add another difference of 1 pixel to the extent of the visual rect), I guess it's safe to assume that this is not an issue for anybody else and it will remain as is.

Then here are the proposed changes to RectTracker, so that ALIGN\_NULL will work:

```
void RectTracker::MouseMove(Point, dword)
{
    Point p = GetMousePos();
    rect = org;
    if(tx == ALIGN_CENTER && ty == ALIGN_CENTER) {
        int x = org.left - op.x + p.x;
        int y = org.top - op.y + p.y;
        if(x + org.Width() > maxrect.right)
            x = maxrect.right - org.Width();
        if(x < maxrect.left)
            x = maxrect.left;
        if(y + org.Height() > maxrect.bottom)
            y = maxrect.bottom - org.Height();
        if(y < maxrect.top)
            y = maxrect.top;
        rect = RectC(x, y, org.Width(), org.Height());
    }
    else {
        if(tx == ALIGN_LEFT) {
            rect.left = max(org.left - op.x + p.x, maxrect.left);
            rect.left = minmax(rect.left, rect.right - maxsize.cx, rect.right - minsize.cx);
        }
        if(tx == ALIGN_RIGHT) {
            rect.right = min(org.right - op.x + p.x, maxrect.right);
            rect.right = minmax(rect.right, rect.left + minsize.cx, rect.left + maxsize.cx);
        }
        if(ty == ALIGN_TOP) {
            rect.top = max(org.top - op.y + p.y, maxrect.top);
            rect.top = minmax(rect.top, rect.bottom - maxsize.cy, rect.bottom - minsize.cy);
        }
        if(ty == ALIGN_BOTTOM) {
            rect.bottom = min(org.bottom - op.y + p.y, maxrect.bottom);
            rect.bottom = minmax(rect.bottom, rect.top + minsize.cy, rect.top + maxsize.cy);
        }
        if(tx == ALIGN_NULL) {
            rect.right = min(org.right - op.x + p.x, maxrect.right);
        }
    }
}
```

```

if(ty == ALIGN_NULL) {
    rect.bottom = min(org.bottom - op.y + p.y, maxrect.bottom);
}
if(keepratio) {
    int cy = org.Width() ? rect.Width() * org.Height() / org.Width() : 0;
    int cx = org.Height() ? rect.Height() * org.Width() / org.Height() : 0;
    if(tx == ALIGN_BOTTOM && ty == ALIGN_RIGHT) {
        Size sz = rect.Size();
        if(cx > sz.cx)
            rect.right = rect.left + cx;
        else
            rect.bottom = rect.top + cy;
    }
    else
        if(tx == ALIGN_RIGHT)
            rect.bottom = rect.top + cy;
        else
            if(ty == ALIGN_BOTTOM)
                rect.right = rect.left + cx;
    }
}
if(rect != o) {
    rect = Round(rect);
    if(rect != o) {
        DrawRect(o, rect);
        sync(rect);
        o = rect;
    }
}
}
}
}

```

```

void RectTracker::DrawRect(Rect r1, Rect r2)
{
    if(ty < 0) {
        r1.left = r1.right - 1;
        r2.left = r2.right - 1;
    }
    if(tx < 0) {
        r1.top = r1.bottom - 1;
        r2.top = r2.bottom - 1;
    }
    Rect c = clip & GetMaster().GetSize();
    if (r1.right < r1.left) Swap(r1.left, r1.right);
    if (r1.bottom < r1.top) Swap(r1.top, r1.bottom);
    if (r2.right < r2.left) Swap(r2.left, r2.right);
    if (r2.bottom < r2.top) Swap(r2.top, r2.bottom);
    if(animation) {

```

```
int nanim = (GetTickCount() / animation) % 8;
DrawDragRect(GetMaster(), Rect(0, 0, 0, 0), r2, c, width, color, sGetAniPat(pattern, nanim));
DrawDragRect(GetMaster(), r1, Rect(0, 0, 0, 0), c, width, color, sGetAniPat(pattern, panim));
panim = nanim;
}
else
DrawDragRect(GetMaster(), r1, r2, c, width, color, pattern);
}
```

---