Subject: Changing tree control style
Posted by gprentice on Sun, 14 Nov 2010 10:43:49 GMT
View Forum Message <> Reply to Message

I dislike the plus/minus icons in the U++ tree control and I would like to have dotted lines instead of solid lines as some window apps do, as in the attached image. As discussed in a previous thread I can see how to change the plus minus images.

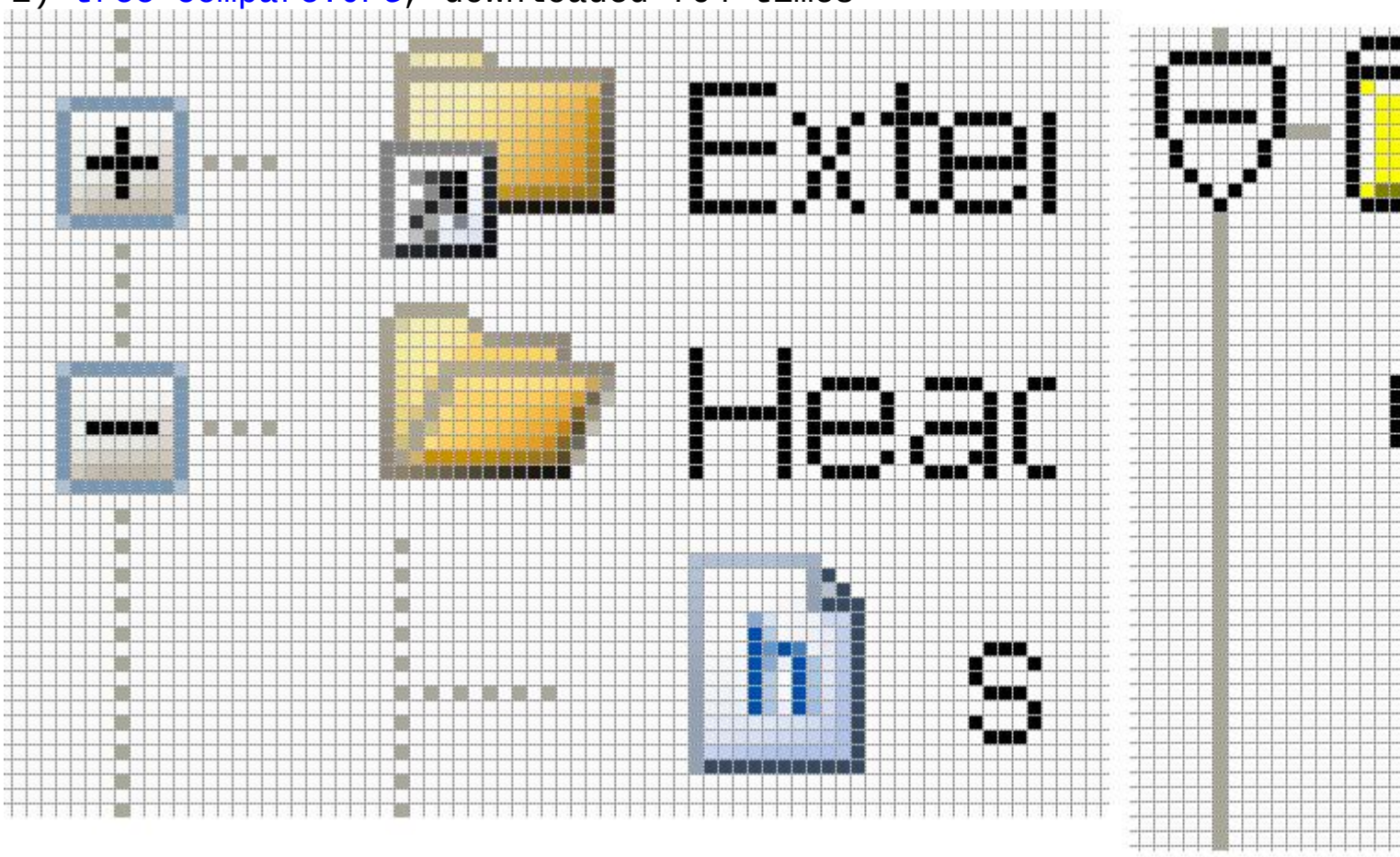
Is all the code that would need to be changed in the TreeCtrl::Paint function? I don't know the U++ library very well yet and I haven't spent enough time to figure out exactly where it's drawing the lines (there's no DrawLine in the Paint function) but I'm wondering if there's a simple change that would give dotted lines instead of solid lines and a slightly different line colour.

If I figure out how to change the code, do I have to maintain my own treectrl.cpp or can an extra style be added to CtrlLib treectrl?

Graeme

File Attachments

1) tree-compare.JPG, downloaded 764 times



Subject: Re: Changing tree control style
Posted by dolik.rce on Sun, 14 Nov 2010 15:57:47 GMT
View Forum Message <> Reply to Message

Hi Graeme

All you need to change is in TreeCtrl::Paint(). There is no DrawLine simply because the lines are drawn using DrawRect with width (or height) of 1. Paint() is virtual, so the easiest way to get the behavior you want is to make an inherited class MyTreeCtrl:public Tree and override its Paint().

The only problem you might encounter is actually how to draw the dotted line, if I remember correctly there is no simple function to draw a line with pattern using Draw. So you will probably have to draw the dots somehow yourself.

Best regards,
Honza

Subject: Re: Changing tree control style
Posted by andrei_natanael on Sun, 14 Nov 2010 19:56:42 GMT
View Forum Message <> Reply to Message

dolik.rce wrote on Sun, 14 November 2010 17:57
The only problem you might encounter is actually how to draw the dotted line, if I remember correctly there is no simple function to draw a line with pattern using Draw. So you will probably have to draw the dots somehow yourself.

There is Draw::DrawLine. 5-th argument may be PEN_SOLID, PEN_DASH, PEN_DOT, PEN_DASHDOT, PEN_DASHDOTDOT if you want to have different kind of lines.

Andrei

Subject: Re: Changing tree control style
Posted by koldo on Sun, 14 Nov 2010 20:37:02 GMT
View Forum Message <> Reply to Message

andrei_natanael wrote on Sun, 14 November 2010 20:56dolik.rce wrote on Sun, 14 November 2010 17:57
The only problem you might encounter is actually how to draw the dotted line, if I remember correctly there is no simple function to draw a line with pattern using Draw. So you will probably have to draw the dots somehow yourself.

There is Draw::DrawLine. 5-th argument may be PEN_SOLID, PEN_DASH, PEN_DOT, PEN_DASHDOT, PEN_DASHDOTDOT if you want to have different kind of lines.

Andrei

Hello Andrei

Those includes only work if Painter is used. If not use a negative color and width == 1 to get a dashed line.

Subject: Re: Changing tree control style
Posted by gprentice on Mon, 15 Nov 2010 10:23:57 GMT
View Forum Message <> Reply to Message

Thanks. I can't use a derived class because the Paint function needs access to the private data of the class to draw the tree. The logical thing is to add a style option and change TreeCtrl.cpp as was talked about in the other thread. If I can figure out how to do it, I'll try and persuade Mirek to accept the code into svn. Anyway, its easy enough to add a style choice without changing much code in the Paint function so there should be no risk to existing apps.

Graeme

Subject: Re: Changing tree control style
Posted by gprentice on Mon, 15 Nov 2010 10:25:43 GMT
View Forum Message <> Reply to Message

koldo wrote on Mon, 15 November 2010 09:37andrei_natanael wrote on Sun, 14 November 2010 20:56dolik.rce wrote on Sun, 14 November 2010 17:57
The only problem you might encounter is actually how to draw the dotted line, if I remember correctly there is no simple function to draw a line with pattern using Draw. So you will probably have to draw the dots somehow yourself.

There is Draw::DrawLine. 5-th argument may be PEN_SOLID, PEN_DASH, PEN_DOT, PEN_DASHDOT, PEN_DASHDOTDOT if you want to have different kind of lines.

Andrei

Hello Andrei

Those includes only work if Painter is used. If not use a negative color and width == 1 to get a dashed line.

What is meant by "if Painter is used" ?

Subject: Re: Changing tree control style
Posted by andrei_natanael on Mon, 15 Nov 2010 11:19:26 GMT
View Forum Message <> Reply to Message

koldo wrote on Sun, 14 November 2010 22:37andrei_natanael wrote on Sun, 14 November 2010 20:56dolik.rce wrote on Sun, 14 November 2010 17:57
The only problem you might encounter is actually how to draw the dotted line, if I remember correctly there is no simple function to draw a line with pattern using Draw. So you will probably have to draw the dots somehow yourself.

There is Draw::DrawLine. 5-th argument may be PEN_SOLID, PEN_DASH, PEN_DOT, PEN_DASHDOT, PEN_DASHDOTDOT if you want to have different kind of lines.

Andrei

Hello Andrei

Those includes only work if Painter is used. If not use a negative color and width == 1 to get a dashed line.

AFAIK it's not true. I've checked code in CtrlCore and it support dashed lines without Painter to be included, see DrawWin32.cpp @ 144 and DrawX11.cpp @ 307.

gprentice wrote
What is meant by "if Painter is used" ?
Painter is a U++ package which support advanced graphics (vectorial). It replace basic Draw if Painter package is included in your project and your graphics looks better . See PainterExamples for more details.

Andrei

Subject: Re: Changing tree control style
Posted by koldo on Mon, 15 Nov 2010 16:05:56 GMT
View Forum Message <> Reply to Message

Quote:AFAIK it's not true. I've checked code in CtrlCore and it support dashed lines without Painter to be included, see DrawWin32.cpp @ 144 and DrawX11.cpp @ 307.
Hello Andrei

Sorry, it is not negative color, it is negative width. See this in DrawWin32.cpp @ 144:void SystemDraw::SetDrawPen(int width, Color color) {
...
```
 if(width != lastPen || color != lastPenColor) {
  static int penstyle[] = {
   PS_NULL, PS_SOLID, PS_DASH,
  #ifndef PLATFORM_WINCE
   PS_DOT, PS_DASHDOT, PS_DASHDOTDOT
```

```
  #endif
  };
  HPEN oldPen = actPen;
  actPen = CreatePen(width < 0 ? penstyle[-width - 1] : PS_SOLID,
                width < 0 ? 0 : width, GetColor(color));
...
 }
}
```

And this in DrawX1132.cpp @ 307, see variable "i":

```
void SystemDraw::SetLineStyle(int width) {
...
 if(width < PEN_SOLID) {
  static const char dash[] = { 18, 6 };
  static const char dot[] = { 3, 3 };
  static const char dashdot[] = { 9, 6, 3, 6 };
  static const char dashdotdot[] = { 9, 3, 3, 3, 3, 3 };
  static struct {
   const char *dash;
   int   len;
  } ds[] = {
   { dash, __countof(dash) },
   { dot, __countof(dot) },
   { dashdot, __countof(dashdot) },
   { dashdotdot, __countof(dashdotdot) }
  };
  int i = -(width - PEN_DASH);
  ASSERT(i >= 0 && i < 4);
  XSetDashes(Xdisplay, gc, 0, ds[i].dash, ds[i].len);
 }
 XSetLineAttributes(Xdisplay, gc, max(width, 1),
                    width < PEN_SOLID ? LineOnOffDash : LineSolid, CapRound, JoinRound);
}
```

A width == -3 means dashed, -4 means dotted, ..., all with a thin line.

Subject: Re: Changing tree control style
Posted by gprentice on Fri, 07 Jan 2011 07:01:54 GMT

View Forum Message <> Reply to Message

To get the dotted line that I want I've had to do a series of DrawRect to draw a one pixel by one pixel rect i.e. a single pixel. The "dotted" line style supported by Windows and U++ is 3 pixels solid, 3 pixels clear which is not what I want. Is there any more efficient way to draw one pixel that calling DrawRect?

From TreeCtrl::Paint

```
        if(yh >= 0 && yl < sz.cy) {
          int x = levelcx + levelcx * l.level + levelcx2 - org.x;
          //w.DrawRect(x, yl, 1, yh - yl, SColorShadow);
```

```
            for (int k = 0; k < (yh - yl); k += 2)
              w.DrawRect(x, yl + k, 1, 1, SColorShadow);

//...

      if(w.IsPainting(0, y, sz.cx, msz.cy) && msz.cy > 0) {
        //w.DrawRect(op.x, op.y, levelcx2, 1, SColorShadow);
        for (int k = 2; k < levelcx2; k += 2)
          w.DrawRect(op.x + k, op.y, 1, 1, SColorShadow);
```