Subject: Bug: Missing width in DrawArc on X11
Posted by steffen on Sat, 17 Mar 2012 21:28:09 GMT
View Forum Message <> Reply to Message

Hi,

I am working on a set of widgets for an embedded touch screen project I recently started. First running on x86, but it should end up running on an Arm9 board currently under development.

I am trying to make some widgets with round corners and ran into a problem with DrawArc.

It seems like SystemDraw::DrawArcOp in DrawOpX11.cpp is not using the width parameter.
I think it is mising a call to SetLineStyle.

When I add the following to my paint function, it works:
((SystemDraw&)w).SetLineStyle(3);

Is there a reason that SetLineStyle is not part of Draw? The type casting seems a bit ugly.

Regards,
Steffen

Edit:
I'm using Upp-4193 on Ubuntu 10.10 64-bit.

Subject: Re: Bug: Missing width in DrawArc on X11
Posted by Tom1 on Fri, 23 Mar 2012 19:44:44 GMT
View Forum Message <> Reply to Message

Hi,

I agree with you: There should definitely be a SetLineStyle() call in the beginning of DrawArcOp() in file DrawOpX11.cpp as follows:

```
void SystemDraw::DrawArcOp(const Rect& rc, Point start, Point end, int width, Color color)
{
 GuiLock __;
 SetLineStyle(width);     // This needs to be added
 XGCValues gcv, gcv_old;
 XGetGCValues(Xdisplay, GetGC(), GCForeground, &gcv_old);
 Point offset = GetOffset();
       ...
```

Hope someone in the core team commits this soon. (I guess the DrawArc is a very little used functionality -- at least on X11 platforms -- since nobody has discovered and fixed it before.)

The SetLineStyle() should not be needed in applications since all drawing primitives should call it as needed for desired results.

Best regards,

Tom

Subject: Re: Bug: Missing width in DrawArc on X11
Posted by Tom1 on Wed, 28 Mar 2012 09:05:27 GMT
View Forum Message <> Reply to Message

Hi,

As it turns out, there is a related issue in DrawOpX11.cpp SystemDraw::DrawPolyPolylineOp and DrawPolyPolyPolygonOp. Neither of them call SetLineStyle for the outline. Instead, both of them use XGetGCValues() and XChangeGC(), but fail to update state of dashing causing unexpected results when used after drawing a dashed line -- or attempting to draw a dashed polyline or polygon-outline.

Using SetLineStyle() in proper places (and skipping GCLineWidth) fixes this issue.

Here are the relevant parts that need to be fixed for Polyline, Polygon and Arc to work properly:

```
void SystemDraw::DrawPolyPolylineOp(const Point *vertices, int vertex_count,
                    const int *counts, int count_count,
                    int width, Color color, Color doxor)
{
 GuiLock __;
 ASSERT(count_count > 0 && vertex_count > 0);
 if(vertex_count < 2 || IsNull(color))
  return;
 SetLineStyle(width);  // Added
 XGCValues gcv_old, gcv_new;
 XGetGCValues(Xdisplay, GetGC(), GCForeground | GCFunction, &gcv_old);
// XGetGCValues(Xdisplay, GetGC(), GCForeground | GCLineWidth | GCFunction, &gcv_old);
 gcv_new.function = IsNull(doxor) ? X11_ROP2_COPY : X11_ROP2_XOR;
 gcv_new.foreground = GetXPixel(color) ^ (IsNull(doxor) ? 0 : GetXPixel(doxor));
// gcv_new.line_width = width;
 XChangeGC(Xdisplay, GetGC(), GCForeground | GCFunction, &gcv_new);
// XChangeGC(Xdisplay, GetGC(), GCForeground | GCLineWidth | GCFunction, &gcv_new);
 enum { REQUEST_LENGTH = 256 }; // X server XDrawLines request length (heuristic)
 Point offset = GetOffset();
 if(vertex_count == 2)
  XDrawLine(Xdisplay, GetDrawable(), GetGC(),
   vertices[0].x + offset.x, vertices[0].y + offset.y,
   vertices[1].x + offset.x, vertices[1].y + offset.y);
```

```
 else if(count_count == 1 || vertex_count > count_count * (REQUEST_LENGTH + 2)) {
  for(; --count_count >= 0; counts++)
  {
   Buffer<XPoint> part(*counts);
   for(XPoint *vo = part, *ve = vo + *counts; vo < ve; vo++, vertices++)
   {
    vo -> x = (short)(vertices -> x + offset.x);
    vo -> y = (short)(vertices -> y + offset.y);
   }
   XDrawLines(Xdisplay, GetDrawable(), GetGC(), part, *counts, CoordModeOrigin);
  }
 }
 else {
  int segment_count = vertex_count - count_count;
  Buffer<XSegment> segments(segment_count);
  XSegment *so = segments;
  while(--count_count >= 0)
  {
   const Point *end = vertices + *counts++;
   so -> x1 = (short)(vertices -> x + offset.x);
   so -> y1 = (short)(vertices -> y + offset.y);
   vertices++;
   so -> x2 = (short)(vertices -> x + offset.x);
   so -> y2 = (short)(vertices -> y + offset.y);
   so++;
   while(++vertices < end) {
    so -> x1 = so[-1].x2;
    so -> y1 = so[-1].y2;
    so -> x2 = (short)(vertices -> x + offset.x);
    so -> y2 = (short)(vertices -> y + offset.y);
    so++;
   }
  }
  ASSERT(so == segments + segment_count);
  XDrawSegments(Xdisplay, GetDrawable(), GetGC(), segments, segment_count);
 }
 XChangeGC(Xdisplay, GetGC(), GCForeground | GCFunction, &gcv_old);
// XChangeGC(Xdisplay, GetGC(), GCForeground | GCLineWidth | GCFunction, &gcv_old);
}

...

void SystemDraw::DrawPolyPolyPolygonOp(const Point *vertices, int vertex_count,
  const int *subpolygon_counts, int subpolygon_count_count,
 const int *disjunct_polygon_counts, int disjunct_polygon_count_count,
 Color color, int width, Color outline, uint64 pattern, Color doxor)
{
 GuiLock __;
```

```
 if(vertex_count == 0)
  return;

 if(!IsNull(outline)) SetLineStyle(width); // Added
 bool is_xor = !IsNull(doxor);
 XGCValues gcv_old, gcv_new;
 XGetGCValues(Xdisplay, GetGC(), GCForeground | GCFunction, &gcv_old);
// XGetGCValues(Xdisplay, GetGC(), GCForeground | GCFunction | GCLineWidth, &gcv_old);
 unsigned xor_pixel = (is_xor ? GetXPixel(doxor) : 0);
 if(!IsNull(color))
 {
  gcv_new.foreground = GetXPixel(color) ^ xor_pixel;
  gcv_new.function = is_xor ? X11_ROP2_XOR : X11_ROP2_COPY;
  XChangeGC(Xdisplay, GetGC(), GCForeground | GCFunction, &gcv_new);
  DrawPolyPolyPolygonRaw(*this, vertices, vertex_count,
   subpolygon_counts, subpolygon_count_count,
   disjunct_polygon_counts, disjunct_polygon_count_count);
 }
 if(!IsNull(outline))
 {
  gcv_new.foreground = GetXPixel(outline) ^ xor_pixel;
  //gcv_new.line_width = width;
  XChangeGC(Xdisplay, GetGC(), GCForeground, &gcv_new);
  //XChangeGC(Xdisplay, GetGC(), GCForeground | GCLineWidth, &gcv_new);
  Point offset = GetOffset();
  for(const int *sp = subpolygon_counts, *se = sp + subpolygon_count_count; sp < se; sp++)
  {
   Buffer<XPoint> segment(*sp + 1);
   XPoint *out = segment;
   for(const Point *end = vertices + *sp; vertices < end; vertices++, out++)
   {
    out -> x = (short)(vertices -> x + offset.x);
    out -> y = (short)(vertices -> y + offset.y);
   }
   *out = segment[0];
   XDrawLines(Xdisplay, GetDrawable(), GetGC(), segment, *sp + 1, CoordModeOrigin);
  }
 }
 XChangeGC(Xdisplay, GetGC(), GCForeground | GCFunction, &gcv_old);
// XChangeGC(Xdisplay, GetGC(), GCForeground | GCFunction | GCLineWidth, &gcv_old);
}


...


void SystemDraw::DrawArcOp(const Rect& rc, Point start, Point end, int width, Color color)
{
```

```
 GuiLock __;
 SetLineStyle(width); // Added
 XGCValues gcv, gcv_old;
 XGetGCValues(Xdisplay, GetGC(), GCForeground, &gcv_old);
 Point offset = GetOffset();
 gcv.foreground = GetXPixel(color);
 XChangeGC(Xdisplay, GetGC(), GCForeground, &gcv);
 Point centre = rc.CenterPoint();
 int angle1 = fround(360 * 64 / (2 * M_PI) *
                     atan2(centre.y - start.y, start.x - centre.x));
 int angle2 = fround(360 * 64 / (2 * M_PI) *
                     atan2(centre.y - end.y, end.x - centre.x));
 if(angle2 <= angle1)
  angle2 += 360 * 64;
 angle2 -= angle1;
 XDrawArc(Xdisplay, GetDrawable(), GetGC(), rc.left + offset.x, rc.top + offset.y,
  rc.Width(), rc.Height(), angle1, angle2);
 XChangeGC(Xdisplay, GetGC(), GCForeground, &gcv_old);
}
```

Best regards,

Tom

Subject: Re: Bug: Missing width in DrawArc on X11
Posted by mirek on Sun, 01 Apr 2012 14:47:25 GMT
View Forum Message <> Reply to Message

Patch applied, thank you.
