Subject: Multi monitor support for X11 backend
Posted by Zbych on Sun, 09 Jun 2013 13:20:29 GMT
View Forum Message <> Reply to Message

Hi,

I would like to improve a little bit multi monitor handling in X11 backend. Right now two (or more) screens are treated as one big display.
We can use Xinerama or Xrandr to get information on screens resolution & position and _NET_WM_STRUT_PARTIAL to mark areas occupied by menus, status bars etc.

Bellow are modified functions:

CtrlCore.upp:

```
library(LINUX !RAINBOW) "X11 Xrender Xinerama";
```

X11Gui.h:

```
#include <X11/extensions/Xrender.h>
#include <X11/extensions/Xinerama.h> //<--- new header
```

X11App.cpp:

```
static Array<Rect> GetScreenArea()
{
 Array<Rect> out;
 int displays;

 XineramaScreenInfo * si = XineramaQueryScreens (Xdisplay, &displays);
 if (si == NULL || displays < 1) out.Add(RectC(0, 0, Xwidth, Xheight));
 else{
  for (int i = 0; i < displays; i++){
   out.Add(RectC(si[i].x_org, si[i].y_org, si[i].width, si[i].height));
  }
  XFree(si);
 }
 return out;
}

void Ctrl::GetWorkArea(Array<Rect>& out)
{
 enum { left = 0, right, top, bottom,
   left_start_y, left_end_y,
   right_start_y, right_end_y,
   top_start_x, top_end_x,
```

```
  bottom_start_x, bottom_end_x
 };

 Array<Rect> sa = GetScreenArea();
 out <<= sa;

 Rect total(sa[0]);
 for (int i = 1; i < sa.GetCount(); i++){
  total |= sa[i];
 }

 Vector<int> wnd_lst = GetPropertyInts(Xroot, XAtom("_NET_CLIENT_LIST"));
 for (int i = 0; i < wnd_lst.GetCount(); i++){
  Vector<int> struts = GetPropertyInts(wnd_lst[i], XAtom("_NET_WM_STRUT_PARTIAL"));
  if (struts.GetCount() != 12) continue;

  for (int j = 0; j < sa.GetCount(); j++){

   if (struts[left] && sa[j].left <= struts[left] && sa[j].right >= struts[left]
    && sa[j].Intersects(Rect(sa[j].left, struts[left_start_y], struts[left], struts[left_end_y])))
    out[j].left = struts[left];

   int tmp = total.right - struts[right];
   if (struts[right] && sa[j].left <= struts[right] && sa[j].right >= struts[right]
    && sa[j].Intersects(Rect(tmp, struts[right_start_y], total.right, struts[right_end_y])))
    out[j].right  = tmp;

   if (struts[top] && sa[j].top <= struts[top] && sa[j].bottom >= struts[top]
    && sa[j].Intersects(Rect(struts[top_start_x], sa[j].top, struts[top_end_x], struts[top])))
    out[j].top    = struts[top];

   tmp = total.bottom - struts[bottom];
   if (struts[bottom] && sa[j].top <= struts[bottom] && sa[j].bottom >= struts[bottom]
    && sa[j].Intersects(Rect(struts[bottom_start_x], tmp, struts[bottom_end_x], total.bottom)))
    out[j].bottom = tmp;
  }
 }
}

Rect Ctrl::GetWorkArea() const
{
 return GetPrimaryWorkArea();
}

Rect Ctrl::GetWorkArea(Point pt)
{
 Array<Rect> rc;
 GetWorkArea(rc);
```

```
 for(int i = 0; i < rc.GetCount(); i++)
  if(rc[i].Contains(pt))
   return rc[i];
 return rc[0];
}

Rect Ctrl::GetVirtualWorkArea()
{
 Array <Rect> wa;
 GetWorkArea(wa);
 Rect r(wa[0]);
 for (int i = 1; i < wa.GetCount(); i++)
  r |= wa[i];

 return r;
}

Rect Ctrl::GetVirtualScreenArea()
{
 Array<Rect> sa = GetScreenArea();

 Rect out(sa[0]);
 for (int i = 1; i < sa.GetCount(); i++){
  out |= sa[i];
 }

 return out;
}

Rect Ctrl::GetPrimaryWorkArea()
{
 Array<Rect> x;
 GetWorkArea(x);
 return x[0];
}

Rect Ctrl::GetPrimaryScreenArea()
{
 return GetScreenArea()[0];
}
```

So far I made test on ATI and Intel graphics. Can someone compile TheIde in NOGTK mode and test window positioning on NVidia card?

Edit:
New version uploaded, should work with bottom and right panels.