High resolution display support in Gnome

Alexander Larsson <alexl@redhat.com>
Who am I?
Problem definition
WTF?

I can't read!
Enter: DPI – Dots Per Inch

- Pixels per inch on the output screen
- Historically around 100 DPI for monitors
- Chromebook Pixel has 239 DPI

- Setting in X (Xft.DPI)
- Normally set to 96 by Gnome
- Tweakable via
  org.gnome.desktop.interface.text-scaling-factor
It only changed the text size! Why?
Why not scale everything with DPI?

For monitors with “traditional” densities:
- One pixel is “visible”
- Rendering needs to match pixel grid to not look fuzzy
- **Even** when the rendering is “vector” based

Also:
- Lots of content is pixel based
  - Icons
  - Pictures
  - Cursors
Choose the order of information to appear in the list view.

- Name
- Size
- Type
- Modified
- Group
- Location
- MIME Type
Choose the order of information to appear in the view.

- Name
- Size
- Type
- Modified

Fuzzy, variable width lines
Scaled pixel sources
Rendered at 120%
Why not scale everything with DPI? *(cont)*

Also, it doesn't really matter in practice:

Height of buttons on screen:
- Laptop screen: 6mm
- 23” external monitor: 8mm

Both work fine.

However, on the Pixel it clearly *doesn't* work
Icons are too small
And don't match text height
Fixed size widgets are hard to hit
Images used are small
Default window sizes are too small
Widget min/default sizes are wrong
Mouse pointer is too small
And it moves too slowly
Hard to resize windows
Not enough padding
Borders too thin
Summing it up

- Lots of places specify sizes in pixels
- Implicit sizes based on images don't scale
- Don't want to change all existing code
- Don't want to make code more complex
Multiple monitors

- External monitors may have lower DPI
- A window can be visible on multiple monitors

Want a window to have *approximately* the same physical size on all monitors.
Solution:

- Specify sizes in an abstract “pixel” size
- On “traditional” monitors these are the same as monitor pixels
- On “HiDPI” monitors these are scaled to multiple monitor pixels
- Use integer scaling factor to keep pixel grid alignment
- Scaling is applied automatically
- Vector-based drawing renders in higher resolution
- Allow specifying high resolution alternatives for pixel data
  - Icons
  - CSS images
How does this affect programmers - Cairo

- cairo_surface_set_device_scale()
- cairo_surface_get_device_scale()
- Automatically applies a scaling factor when rendering
  - Can't unset with e.g. cairo_identity_matrix()
- Applies scale factor when using the surface as source
- You don't really have to care, Gtk+ hides this
How does this affect programmers - Gdk

- **Window, Screen, Monitor:**
  - Sizes reported in abstract pixels
  - Position reported in abstract pixels

- `gdk_window_get_scale_factor()`
  - May change over time

- `gdk_screen_get_monitor_scale_factor()`
  - May change over time

- Mouse position reported in abstract pixels
  - Uses floating point, so has full resolution
How does this affect programmers – Gdk (cont)

- **gdk_window_create_similar_surface()**
  - Creates scaled offscreen surfaces if the window is scaled
  - Use this for all kinds of double buffering to avoid pixelized results
  - Recreate double buffers when scale changes

- **gdk_window_create_similar_image_surface()**
  - Allows creation of scaled image surfaces
  - Lets us specify pixel data which will automatically scale correctly when drawn
  - Allows you to specify the target GdkWindow ahead of time for more efficient rendering

- **gdk_cairo_surface_create_from_pixbuf()**
How does this affect programmers - Gtk

- Size allocation happens in abstract pixels
- GtkWidget::scale
  - Use property notification to detect changes
- cairo_surface_t is the primary way to specify pixel data
  - gtk_image_[new|set]_from_surface
  - GtkWidget::surface
  - Automatic scaling wrt the target scale
  - More efficient to render
- GtkIconTheme supports choosing scaled icons
  - Icon theme spec extension for Scale
  - Otherwise picks larger icons
- CSS extension to specify alternative images:
  - -gtk-scaled(url('file.png'),url('file@2.png'))
  - Make sure they have the same size (sans scale)
Implementation – Wayland

- Protocol additions in Wayland 1.2
- Output/Surface positions in abstract pixels
- Compositor chooses the scale for each output
  - Exposed to client by an Output property
- Clients can chose to supply a buffer of a larger scale
- Compositor scales client buffers as needed
- Allows mixed-scale monitor
  - Including windows displaying on multiple monitors
- Gtk+ picks maximum scale of all monitors the window is on
  - Changes when the window moves between monitors
- Uses cairo surface device-scale to implement scaling
- This is the future!
Implementation – X11

- X screens/monitor size/positions are reported scaled down
- XWindows are larger than the corresponding GdkWindows
- Event coordinates are scaled when converting from X
- Only one scale can be used for all monitors
- GDK_SCALE environment variable
  - Useful for testing
- New Xsetting for scale
  - Will be set automatically in Gnome
- New Xsetting for the “unscaled dpi”
  - Normal dpi can be scaled for non-scale-aware apps
- Uses cairo surface device-scale to implement scaling
- Needs care when mixing with native X/GL operations
Implementation – OSX
Limitations
Thank brion
Future plans