Hi-DPI

Hi-dpi support means that applications render at half the available screen resolution to avoid content that is defined in terms of pixels from becoming tiny. Effectively, this means treating 2x2 blocks as device pixels as application pixels, with the extra twist that data that is available in high resolution (e.g. svgs or fonts) can be rendered at the full resolution.

In theory, it can vary from monitor to monitor on a multi-monitor system whether the conditions for hi-dpi scaling are met. Under X11, GNOME currently supports only a single, global scale factor. For Wayland, we plan to have per-monitor scaling.

Manually enabling Hi-DPI support

GNOME currently enables hi-dpi support when the screen resolution is at least 192 dpi and the screen height (in device pixels) is at least 1200.

There are several ways to manually override this, and force hi-dpi support on or off.

To just launch a single GTK+ application in a forced hi-dpi mode, you can set the GDK_SCALE environment variable to either 1 or 2. Clutter uses the environment variable CLUTTER_SCALE for the same purpose. You may need to set both for applications that use GTK+ and clutter.

The GTK+ Inspector has a control in its Visual tab for changing the scale factor at runtime.

To force hi-dpi mode for the entire desktop, gnome-tweak-tool has a control for the scale factor in its Windows tab. You can also override the XSetting manually, using

```
gsettings set org.gnome.settings-daemon.plugins.xsettings overrides "[{'Gdk/WindowScalingFactor', <2>}]"
```

Note that the desktop-wide changes require gnome-settings-daemon to be running.

Hi-DPI for developers

GTK+ makes hi-dpi work transparently in most places, but sometimes it may be necessary to do some extra work for best results. Here are some APIs to learn about the scale factor:

```
gdk_screen_get_monitor_scale_factor
gdk_window_get_scale_factor
gtk_widget_get_scale_factor
```

Similarly, Clutter tries to automatically handle the scaling factor on each drawing surface, but it provides explicit API to retrieve and set the scaling factor:

```
ClutterSettings:window-scaling-factor
clutter_canvas_set_scale_factor
clutter_canvas_get_scale_factor
```

Testing Hi-DPI application on Wayland without actual Hi-DPI hardware

In order to test whether an application works in a Hi-DPI setup, but without having the hardware available to test "for real" one can use a nested Wayland compositor that is configured to fake a HiDPI system.

To test an application on a single Hi-DPI monitor, run the following commands to start a nested Wayland compositor:

```
MUTTER_DEBUG_DUMMY_MONITOR_SCALES=2 mutter --wayland --nested
```

To then run your application in the nested Wayland compositor, run the following command:

```
WAYLAND_DISPLAY=wayland-1 ./path/to/application/to/test
```

This will start the application inside the nested Wayland compositor. Note that "wayland-1" may be something else, depending on how many compositors are running already. "wayland-1" means the second compositor. If you run this from within an X session, it will probably be "wayland-0".

In order to test your application on a multi monitor setup, where monitors have different scales, run the following commands to emulate a system where you have two monitors where the left most monitor is Hi-DPI, while the right most has classic DPI.

```
MUTTER_DEBUG_NUM_DUMMY_MONITORS=2 MUTTER_DEBUG_DUMMY_MONITOR_SCALES=2,1 mutter --wayland --nested
```

Run your application in the same way as described above to start it inside the nested Wayland compositor.