Picom (Compton) Configuration

1 Introduction

Picom is the successor to Compton, a standalone compositor for Xorg. It provides compositing for WM that do not provide any, such

2 Shadows

The following enables client-side shadows on windows. Note desktop windows (windows with `_NET_WM_WINDOW_TYPE_DESKTOP`) never get shadow, unless explicitly requested using the wintypes option. I personally deactivated shadows because they don't work out too well with rounded corners.

```
shadow = false;
```

The blur radius radius for shadows is measured in pixels, and it defaults to 12px.

```
shadow-radius = 7;
```

Picom can also apply some level of opacity on shadows.

Default value	`0.75`
Min value	`0.0`
Max value	`1.0`

```
shadow-opacity = 0.85
```

The left and top offsets for shadows are expressed in pixels.

Default value	`-15`

The following values have an impact on the shadow's RGB color.

Default value	`0.0`
Min value	`0.0`
Max value	`1.0`

```
shadow-red = 0.0;
shadow-green = 0.0;
shadow-blue = 0.0;
```

It is possible to specify a list of conditions of windows that should have no shadow.

Default value	`[]`

```
shadow-exclude = [
  "name = 'Notification'",
  "class_g = 'Conky'",
  "class_g ?= 'Notify-osd'",
  "class_g = 'Cairo-clock'",
  "_GTK_FRAME_EXTENTS@:c"
];
```

It is also possible to specify an X geometry that describes the region in which shadows should not be painted in, such as a dock window region. For example,

```
# shadow-exclude-reg = "x10+0+0"
```

would make the 10 pixels at the bottom of the screen not have any

Default value	""

```
shadow-exclude-reg = ""
```

Finally, it is also possible to crop the shadow of a window fully on a particular Xinerama screen to the screen.

Default value

`false`

```
xinerama-shadow-crop = false
```

2.1 Deprecated options

Options in this subheader **will not** be exported to my configuration file.

Thanks to this value, Picom can avoid drawing shadows on dock or panel windows. This option is deprecated, and users should use the `wintypes` option in their config file instead.

Default value	`false`

```
no-dock-shadow = false;
```

This option allows Picom not to draw on drag-and-drop windows. This option is deprecated, and users should use the `wintypes` option in their config file instead.

```
no-dnd-shadow = false;
```

`shadow-ignore-shaped` is also deprecated. It used to indicate Picom not to paint shadows on shaped windows. Note shaped windows here means windows setting their shape through X Shape extension. Those using ARGB background are beyond Picom's control. Since it is deprecated, you could instead use

```
shadow-exclude = 'bounding_shaped'
```

or

```
shadow-exclude = 'bounding_shaped && !rounded_corners'
```

Default value	""

```
shadow-ignore-shaped = ""
```

3 Rounded corners

A great feature added by ibhagwan's fork of compton is the addition of rounded corners from sdhand's fork, and the Kawase blur (described *here*) from tryone144's fork. Here we can see the declaration of the corners' radius:

```
corner-radius = 9.0;
```

It is also possible to exclude some windows from getting their

```
rounded-corners-exclude = [
  "_NET_WM_WINDOW_TYPE@[0]:a = '_NET_WM_WINDOW_TYPE_DOCK'"
];
```

4 Fading

Picom has the ability to create some fading effects on windows when opening or closing or when the opacity changes. The following parameter toggles this feature on or off. However, its behavior can be changed with `no-fading-openclose`.

Default value	`false`

```
fading = true
```

These values controls the opacity change between steps while fading in and out.

Default value	`0.028` (fade-in), `0.03` (fade-out)
Min value	`0.01`
Max value	`1.0`

```
fade-in-step = 0.09;
fade-out-step = 0.08;
```

This value represents the time between steps in fade steps, in milliseconds.

Min value	1

```
fade-delta = 20;
```

It is possible to exclude some windows that should not be faded with a specified list of conditions.

Default value	`[]`

```
fade-exclude = [ "class_g = 'mpv'" ];
```

This option allows Picom not to create any fade on windows opening or closing.

Default value	`false`

```
no-fading-openclose = true;
```

Finally, this option is a workaround for Openbox, Fluxbox and others by not fading destroyed ARGB windows with WM frame.

Default value	`false`

```
no-fading-destroyed-argb = false
```

5 Transparency and

Picom is also able to create some opacity or transparency for windows, depending on their state or on some user-defined rules. For instance, the following value describes the opacity of inactive windows.

Default value	`1.0`
Min value	`0.1`
Max value	`1.0`

```
inactive-opacity = 0.6;
```

On the other hand, it is possible to declare a default opacity for active windows.

Default value	`1.0`
Min value	`0.1`
Max value	`1.0`

```
active-opacity = 1;
```

This however describes the opacity of window titlebars and borders.

Default value	`1.0`
Min value	`0.1`

```
frame-opacity = 1.0;
```

`menu-opacity` describes the opacity for dropdown menus and popup menus.

Default value	`1.0`
Min value	`0.1`
Max value	`1.0`

```
# menu-opacity = 0.9;
```

`inactive-opacity-override` allows the user to let inactive opacity set by `-i` override the ~_NET_WM_OPACITY_ values of windows.

Default value	`true`

```
inactive-opacity-override = true;
```

While it is possible to alter opacity on inactive windows, it is also possible to dim them.

Default value	`1.0`
Min value	`0.1`
Max value	`1.0`

```
# inactive-dim = 1.0
```

Default value	`1.0`
Min value	`0.1`
Max value	`1.0`

```
# inactive-dim-fixed = 1.0
```

It is also possible to specify a list of conditions of windows that should always be considered focused.

Default value	`[]`

```
focus-exclude = [ "class_g = 'mpv'" ];
```

The user can also specify a list of opacity rules, in the format `PERCENT:PATTERN`, like `50:name *= "Firefox"`. picom-trans is recommended over this. Note we don't make any guarantee about possible conflicts with other programs that set `_NET_WM_WINDOW_OPACITY` on frame or client windows.

Default value	`[]`

```
opacity-rule = [
  "85:class_g = 'Polybar'",
  # "55:class_g *?= 'Rofi'",
];
```

6 Background blurring

```
blur: {
  method = "dual_kawase";
  strength = 7;
  background = false;
  background-frame = false;
  background-fixed = false;
}
```

This value enables or disables the blur for the background of semi-transparent or ARGB windows. It has bad performances though, with driver-dependent behavior. The name of the switch may change without prior notifications.

Default value	`false`

```
blur-background = true;
```

Blur background of windows when the window frame is not opaque. If true, this implies the value `true` for `blur-background`.

Default value	`false`

```
blur-background-frame = true;
```

The following determines whether to use fixed blur strength rather than adjusting according to window opacity.

Default value	`false`

```
blur-background-fixed = false;
```

Default value	`""`

```
# blur-kern = "3x3box";
blur-kern = "5,5,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
```

It is possible to write exclude conditions for background blur.

Default value	`[]`

```
blur-background-exclude = [
  "window_type = 'desktop'",
  "class_g = 'Polybar'",
  "_GTK_FRAME_EXTENTS@:c"
];
```

7 General settings

Daemonize process. Fork to background after initialization. Causes issues with certain (badly-written) drivers.

Default value	`false`

```
daemon = true;
```

Picom has three backends it can use: `xrender`, `glx`, and `xr_glx_hybrid`. GLX backend is typically much faster but depends on a sane driver.

```
backend = "glx";
```

This enables or disables VSync.

Default value	`false`

```
vsync = true;
```

Enable remote control via D-Bus. See the *D-BUS API* section below for more details.

Default value	`false`

```
dbus = false;
```

Try to detect WM windows (a non-override-redirect window with no child that has `WM_STATE`) and markz them as active.

Default value	`false`

```
mark-wmwin-focused = true;
```

Mark override-redirect windows that doesn't have a child window with `WM_STATE` focused.

Default value	`false`

```
mark-ovredir-focused = true;
```

Default value	`false`

```
detect-rounded-corners = true;
```

Detect `_NET_WM_OPACITY` on client windows, useful for window managers not passing `_NET_WM_OPACITY` of client windows to frame windows.

Default value	`false`

```
detect-client-opacity = true;
```

Specify refresh rate of the screen. If not specified or 0, picom will try detecting this with X RandR extension.

Default value	`60`

```
refresh-rate = 60;
```

Limit picom to repaint at most once every 1 / `refresh_rate` second to boost performance. This should not be used with

```
vsync drm/opengl/opengl-oml
```

as they essentially does sw-opti's job already, unless you wish to specify a lower refresh rate than the actual value.

Default value	`""`

Use EWMH `_NET_ACTIVE_WINDOW` to determine currently focused window, rather than listening to `FocusIn~/~FocusOut` event. Might have more accuracy, provided that the WM supports it.

Default value	`false`

```
# use-ewmh-active-win = false;
```

Unredirect all windows if a full-screen opaque window is detected, to maximize performance for full-screen windows. Known to cause flickering when redirecting/unredirecting windows. paint-on-overlay may make the flickering less obvious.

Default value	`false`

```
unredir-if-possible = false;
```

Delay before unredirecting the window, in milliseconds.

Default value	`0`

```
unredir-if-possible-delay = 0;
```

Conditions of windows that shouldn't be considered full-screen for unredirecting screen.

Default value	`[]`

```
unredir-if-possible-exclude = [];
```

Default value	`false`

```
detect-transient = true;
```

Use `WM_CLIENT_LEADER` to group windows, and consider windows in the same group focused at the same time. `WM_TRANSIENT_FOR` has higher priority if detect-transient is enabled, too.

Default value	`false`

```
detect-client-leader = true;
```

Resize damaged region by a specific number of pixels. A positive value enlarges it while a negative one shrinks it. If the value is positive, those additional pixels will not be actually painted to screen, only used in blur calculation, and such. (Due to technical limitations, with use-damage, those pixels will still be incorrectly painted to screen.) Primarily used to fix the line corruption issues of blur, in which case you should use the blur radius value here (e.g. with a 3x3 kernel, you should use `--resize-damage 1`, with a 5x5 one you use `--resize-damage 2`, and so on). May or may not work with **–glx-no-stencil**. Shrinking doesn't function correctly.

Default value	`1`

```
resize-damage = 1;
```

Specify a list of conditions of windows that should be painted with inverted color. Resource-hogging, and is not well tested.

```
invert-color-include = [];
```

Disable the use of damage information. This cause the whole screen to be redrawn everytime, instead of the part of the screen has actually changed. Potentially degrades the performance, but might fix some artifacts. The opposing option is use-damage

Default value	`false`

```
use-damage = false;
```

Use X Sync fence to sync clients' draw calls, to make sure all draw calls are finished before picom starts drawing. Needed on nvidia-drivers with GLX backend for some users.

Default value	`false`

```
xrender-sync-fence = false;
```

Force all windows to be painted with blending. Useful if you have a glx-fshader-win that could turn opaque pixels transparent.

Default value	`false`

```
force-win-blend = false;
```

Do not use EWMH to detect fullscreen windows. Reverts to checking if a window is fullscreen based only on its size and coordinates.

```
no-ewmh-fullscreen = false;
```

Dimming bright windows so their brightness doesn't exceed this set value. Brightness of a window is estimated by averaging all pixels in the window, so this could comes with a performance hit. Setting this to 1.0 disables this behaviour. Requires `--use-damage` to be disabled.

Default value	`1.0`

```
max-brightness = 1.0;
```

Make transparent windows clip other windows like non-transparent windows do, instead of blending on top of them.

Default value	`false`

```
transparent-clipping = false;
```

Set the log level. Possible values are:

- `trace`
- `debug`
- `info`
- `warn`
- `error`

in increasing level of importance. Case doesn't matter. If using the "TRACE" log level, it's better to log into a file using `--log-file`, since it can generate a huge stream of logs.

Default value	`"debug"`

Set the log file. If `--log-file` is never specified, logs will be written to stderr. Otherwise, logs will to written to the given file, though some of the early logs might still be written to the stderr. When setting this option from the config file, it is recommended to use an absolute path.

Default value	`''`

```
# log-file = '/path/to/your/log/file';
```

Show all X errors (for debugging)

Default value	`false`

```
# show-all-xerrors = false;
```

Write process ID to a file.

Default value	`''`

```
# write-pid-path = '/path/to/your/log/file';
```

Window type settings. `WINDOW_TYPE` is one of the 15 window types defined in EWMH standard:

- `"unknown"`
- `"desktop"`
- `"dock"`
- `"toolbar"`
- `"menu"`
- `"utility"`

- "dropdown_menu"
- "popup_menu"
- "tooltip"
- "notification"
- "combo"
- "dnd"

Following per window-type options are available:

fade, shadow
: Controls window-type-specific shadow and fade settings.

opacity
: Controls default opacity of the window type.

focus
: Controls whether the window of this type is to be always considered focused. (By default, all window types except "normal" and "dialog" has this on.)

full-shadow
: Controls whether shadow is drawn under the parts of the window that you normally won't be able to see. Useful when the window has parts of it transparent, and you want shadows in those areas.

redir-ignore
: Controls whether this type of windows should cause screen to become redirected again after been unredirected. If you have unredir-if-possible set, and doesn't want certain window to cause unnecessary screen redirection, you can set this to `true`.

```
wintypes:
{
  tooltip = { fade = true; shadow = true; opacity = 0.75; foc
  dock = { shadow = false; }
  dnd = { shadow = false; }
  popup_menu = { opacity = 0.8; }
  dropdown_menu = { opacity = 0.8; }
```

7.1 GLX backend-specific options

Avoid using stencil buffer, useful if you don't have a stencil buffer. Might cause incorrect opacity when rendering transparent content (but never practically happened) and may not work with blur-background. Tests show a 15% performance boost. Recommended.

Default value	`false`

```
glx-no-stencil = true;
```

Avoid rebinding pixmap on window damage. Probably could improve performance on rapid window content changes, but is known to break things on some drivers (LLVMpipe, xf86-video-intel, etc.). Recommended if it works.

Default value	`false`

```
glx-no-rebind-pixmap = false;
```

Use specified GLSL fragment shader for rendering window contents. See `compton-default-fshader-win.glsl` and `compton-fake-transparency-fshader-win.glsl` in the source tree for examples.

Default value	`''`

```
glx-fshader-win = '';
```