The GNOME™ Conference

GUADEC

Reducing power and bandwidth use in apps to keep users happy

Philip Withnall
Endless
philip@tecnocode.co.uk

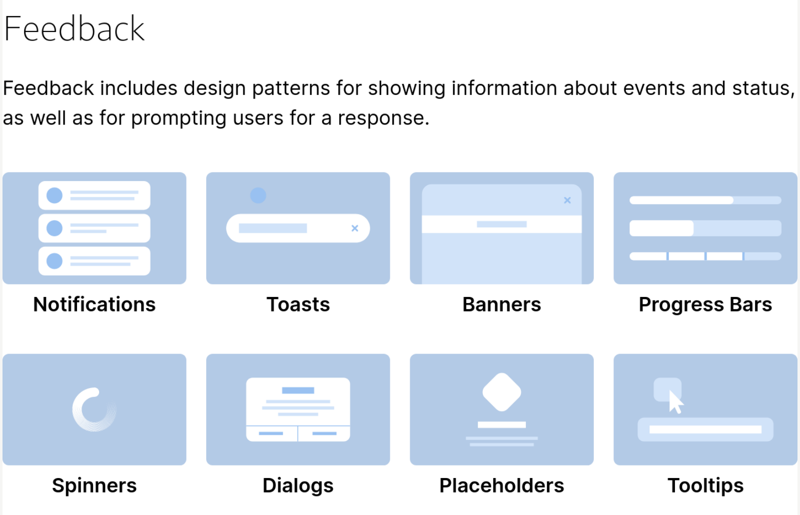
Networks

- Networks can be metered
- Metered status is broadcast by the network, or set by the user
- Metered networks are generally slow or cost money per unit data

Metered API

```
Gio.NetworkMonitor.network-metered
```

Metered UI considerations



Feedback

Feedback includes design patterns for showing information about events and status, as well as for prompting users for a response.

Notifications

Toasts

Banners

Progress Bars

Spinners

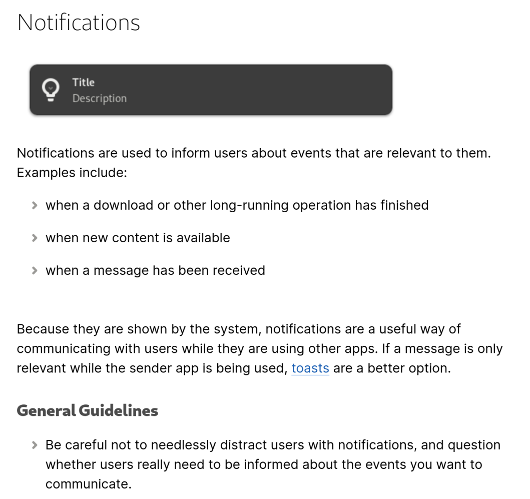
Dialogs

Placeholders

Tooltips

Figure: The HIG page on UI feedback patterns

Metered UI considerations



Notifications

Notifications are used to inform users about events that are relevant to them. Examples include:

- when a download or other long-running operation has finished
- when new content is available
- when a message has been received

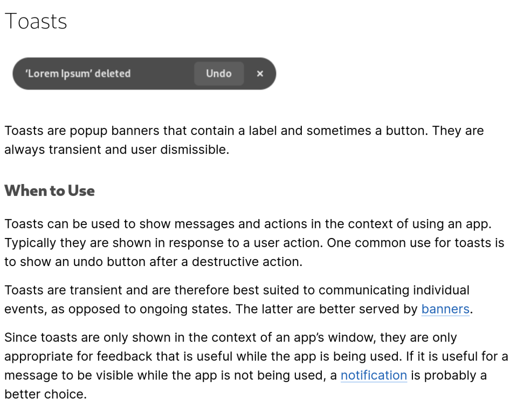
Because they are shown by the system, notifications are a useful way of communicating with users while they are using other apps. If a message is only relevant while the sender app is being used, toasts are a better option.

General Guidelines

- Be careful not to needlessly distract users with notifications, and question whether users really need to be informed about the events you want to communicate.

Figure: The HIG page on notifications

Metered UI considerations



Toasts

Toasts are popup banners that contain a label and sometimes a button. They are always transient and user dismissible.

When to Use

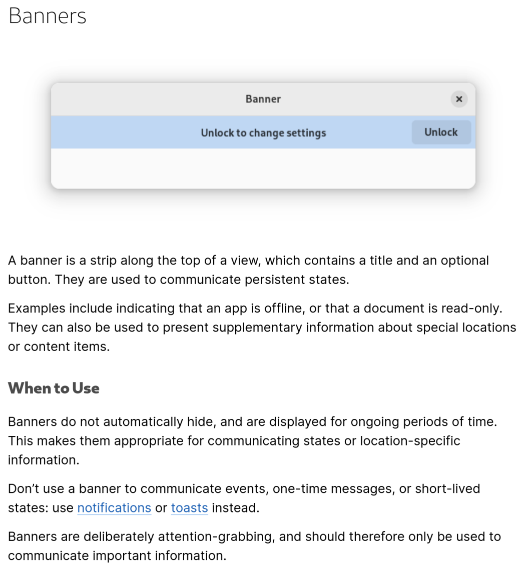
Toasts can be used to show messages and actions in the context of using an app. Typically they are shown in response to a user action. One common use for toasts is to show an undo button after a destructive action.

Toasts are transient and are therefore best suited to communicating individual events, as opposed to ongoing states. The latter are better served by banners.

Since toasts are only shown in the context of an app's window, they are only appropriate for feedback that is useful while the app is being used. If it is useful for a message to be visible while the app is not being used, a notification is probably a better choice.

Figure: The HIG page on toasts

Metered UI considerations



Banners

A banner is a strip along the top of a view, which contains a title and an optional button. They are used to communicate persistent states.

Examples include indicating that an app is offline, or that a document is read-only. They can also be used to present supplementary information about special locations or content items.

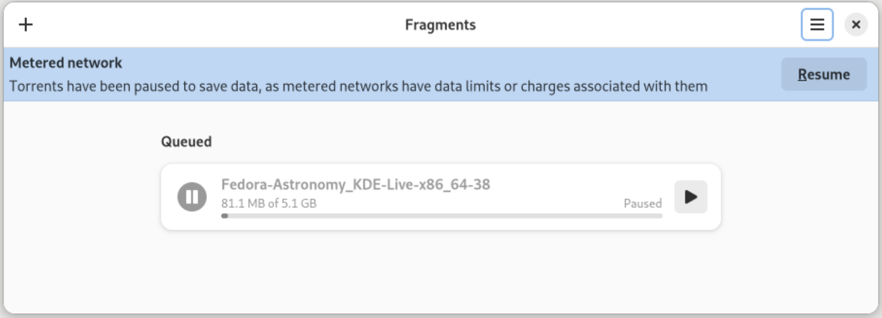
When to Use

Banners do not automatically hide, and are displayed for ongoing periods of time. This makes them appropriate for communicating states or location-specific information.

Don’t use a banner to communicate events, one-time messages, or short-lived states: use notifications or toasts instead.

Banners are deliberately attention-grabbing, and should therefore only be used to communicate important information.

Metered example: Fragments



Metered network

Torrents have been paused to save data, as metered networks have data limits or charges associated with them

Queued

Fedora-Astronomy_KDE-Live-x86_64-38

81.1 MB of 5.1 GB

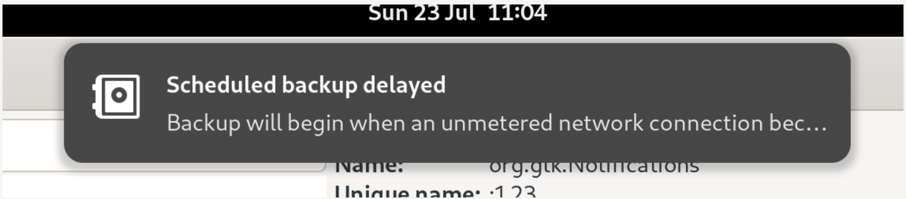
Paused

Figure: Fragments showing an infobar on a metered network

Metered example: Fragments

- `https://gitlab.gnome.org/World/Fragments/-/merge_requests/156`
- Pause torrents when moving onto a metered network
- Resume them when moving off the metered network (unless the user has modified them since)
- Display an infobar while metered

Metered example: Déjà Dup



Sun 23 Jul 11:04

Scheduled backup delayed

Backup will begin when an unmetered network connection bec…

Name: org.gtk.Notifications

Unique name: :1.23

Figure: Déjà Dup showing a notification when on a metered network

Metered example: Déjà Dup

- `https://gitlab.gnome.org/World/deja-dup/-/blob/main/libdeja/Network.vala#L59`
- Doesn't begin a backup while on a metered network
- …unless a preference is set to override this

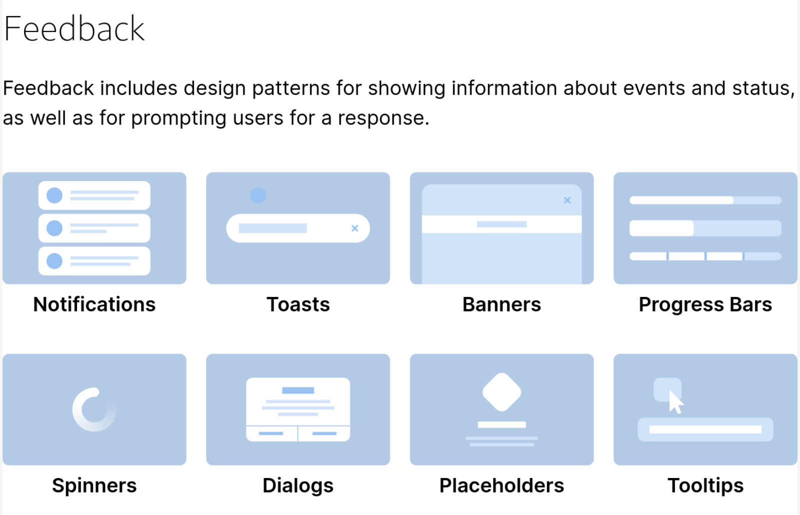
Power

- Traditionally restricted to ‘is the computer on battery?’
- Now a wider definition: power saver mode
- Can be explicitly enabled by the user in anticipation of being away from power for a long time

Power API

```
Gio.PowerProfileMonitor.power-saver-enabled
```

Power UI considerations



Feedback

Feedback includes design patterns for showing information about events and status, as well as for prompting users for a response.

Notifications

Toasts

Banners

Progress Bars

Spinners

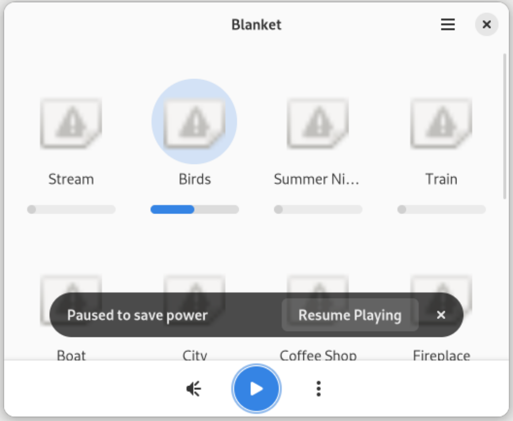
Dialogs

Placeholders

Tooltips

Figure: The HIG page on UI feedback patterns

Power example: Blanket



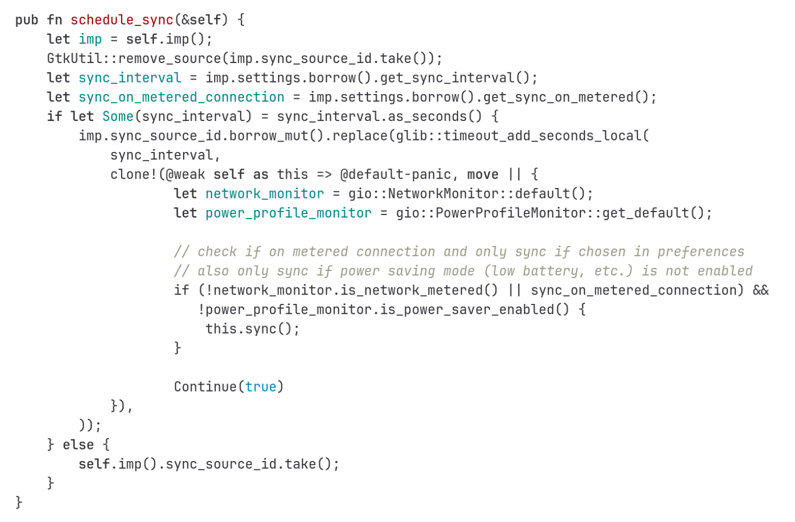
Paused to save power

Figure: Blanket showing a toast when pausing sounds in power-saver mode

Power example: Blanket

- `https://github.com/rafaelmardojai/blanket/pull/312`
- Pauses background playback when entering power saver mode
- Shows a toast explaining why playback has been paused
- In case Blanket's UI is not visible, this toast doesn't have a timeout

Power example: NewsFlash



```rust
pub fn schedule_sync(&self) {
    let imp = self.imp();
    GtkUtil::remove_source(imp.sync_source_id.take());
    let sync_interval = imp.settings.borrow().get_sync_interval();
    let sync_on_metered_connection = imp.settings.borrow().get_sync_on_metered();
    if let Some(sync_interval) = sync_interval.as_seconds() {
        imp.sync_source_id.borrow_mut().replace(glib::timeout_add_seconds_local(
            sync_interval,
            clone!(@weak self as this => @default-panic, move || {
                    let network_monitor = gio::NetworkMonitor::default();
                    let power_profile_monitor = gio::PowerProfileMonitor::get_default();

                    // check if on metered connection and only sync if chosen in preferences
                    // also only sync if power saving mode (low battery, etc.) is not enabled
                    if (!network_monitor.is_network_metered() || sync_on_metered_connection) &&
                       !power_profile_monitor.is_power_saver_enabled() {
                        this.sync();
                    }

                    Continue(true)
            }),
        ));
    } else {
        self.imp().sync_source_id.take();
    }
}
```

Figure: The code from NewsFlash to disable background syncing

Power example: NewsFlash

- `https://gitlab.com/news-flash/news_flash_gtk/-/merge_requests/148`
- Affects whether to enable background syncing of news feeds
- Adds to existing logic which disables them on metered networks
- No UI

What haven't I covered?

- Profile your apps for power usage: reduce wakeups, reduce spinning
- `powertop` is amazing
- Profile your apps for network usage too: `nethogs`
- `Gio.MemoryMonitor.low-memory-warning`
- Don't use web views or write your app using web technologies if you can help it, they're terrible for power use
- Occasionally look at your project's CI resource and GitLab disk use

Miscellany

Metered data initiative
`https://gitlab.gnome.org/GNOME/Initiatives/-/issues/42`

Power-saver initiative
`https://gitlab.gnome.org/GNOME/Initiatives/-/issues/43`

Low memory initiative
`https://gitlab.gnome.org/GNOME/Initiatives/-/issues/44`

Slide source `https://gitlab.gnome.org/pwithnall/guadec-power-bandwidth-presentation-2023`

Beamer theme: `https://gitlab.gnome.org/GNOME/presentation-templates/tree/master/GUADEC/2023`