Snapping like hell(sworth)!

Heather Ellsworth <heather.ellsworth@canonical.com>

Dani Llewellyn <diddledani@ubuntu.com>

Ubuntu Summit - November 2022

lalala

What you will learn:

- What the hellsworth are snaps ;)
- How to snap GNOME applications with a few different examples

Setup:

Create working dir:

```
~$ mkdir snap-examples && cd snap-examples
```

lala

Example #1

Hello World

Gtk Hello World - What are we even doing?

What to expect:

- Use GNOME's Gtk Hello World example to compile binary
- Step-by-step process crating snapcraft.yaml
- Iterative build

Key Concepts:

- Basic metadata, apps, parts
- How to include runtime dependencies (stage-packages)
- Dump plugin
- Local source
- build and test

Gtk Hello World - Make the App

How to make the hello world: https://www.gtk.org/docs/getting-started/hello-world

Setup working directory

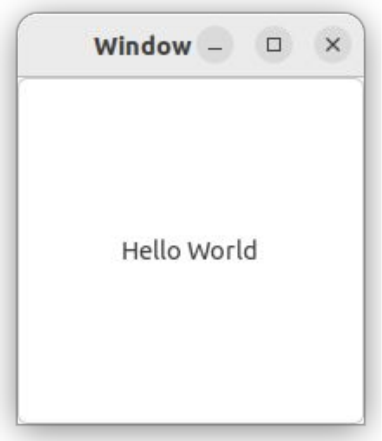
```
~/snap-examples$ mkdir -p hello-world-gtk/src && cd hello-world-gtk/src
```

Copy the script into `hello-world-gtk.c`, and compile the binary:

```
~/snap-examples/hello-world-gtk/src$ gcc $(pkg-config --cflags gtk4) -o hello-world-gtk
hello-world-gtk.c $(pkg-config --libs gtk4)
```

Test the example

```
~/snap-examples/hello-world-gtk/src$ ./hello-world-gtk
```



(Quick grab: https://github.com/hellsworth/hello-world-gtk)

Gtk Hello World - Let's snap it!

We have source + binary

Using your favorite editor, create snapcraft.yaml:

```
~/snap-examples/hello-world-gtk/src$ cd ..

~/snap-examples/hello-world-gtk$ vim snapcraft.yaml
```

Ok what do we need?

- metadata: name, version, summary, description, base, confinement
- apps: list of apps that will be built
- parts: the stuff that builds the apps

(Quick grab: https://github.com/hellsworth/hello-world-gtk)

Gtk Hello World - Metadata

```
~/snap-examples/hello-world-gtk$ vim snapcraft.yaml
```

```
name: hello-world-gtk
version: '0.1'
summary: Gtk Hello World example
description: A simple Gtk example
base: core22
confinement: strict
```

Recommended metadata: https://snapcraft.io/docs/adding-global-metadata

Complete list of metadata keys/values: https://snapcraft.io/docs/snapcraft-top-level-metadata

(Quick grab: https://github.com/hellsworth/hello-world-gtk)

Gtk Hello World - Apps

```
~/snap-examples/hello-world-gtk$ vim snapcraft.yaml
```

```
apps:
  hello-world-gtk:
      command: src/hello-world-gtk
      plugs:
        - x11
        - wayland
```

(Quick grab: https://github.com/hellsworth/hello-world-gtk)

Gtk Hello World - Parts

```
~/snap-examples/hello-world-gtk$ vim snapcraft.yaml


parts:
  hello-world-gtk:
      plugin: dump
      source: .
      stage-packages:
        - libgtk-4-1
```

For a complete list of metadata keys/values: https://snapcraft.io/docs/adding-parts

(Quick grab: https://github.com/hellsworth/hello-world-gtk)

Gtk Hello World - Build it!

Build command: “snapcraft” with options

- v = print stdout

- - debug = in case of failure, drop us into build environment

- - shell = in case of pass, drop us into build environment

```
~/snap-examples/hello-world-gtk$ snapcraft -v --debug
Starting Snapcraft 7.2.1.post14+gitc7ce663f
Logging execution to '/home/heather/.cache/snapcraft/log/snapcraft-20221106-143232.095152.log'
Running on amd64 for amd64
Launching instance...
    <snip>
Creating snap package...
Created snap package hello-world-gtk_0.1_amd64.snap
```

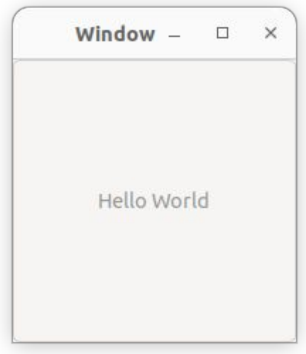
(Quick grab: https://github.com/hellsworth/hello-world-gtk)

Gtk Hello World - Install and Run it!

Build command: “snap” with options

- - dangerous = if not installed from store, circumvent store signature

```
~/snap-examples/hello-world-gtk$ sudo snap install hello-world-gtk_0.1_amd64.snap --dangerous

hello-world-gtk 0.1 installed



$ snap run hello-world-gtk

(hello-world-gtk:109667): Gtk-WARNING **: 14:50:30.500: Locale not supported by C library.

      Using the fallback 'C' locale.

Gsk-Message: 14:50:30.710: Failed to realize renderer of type 'GskGLRenderer' for surface
'GdkX11Toplevel': libEGL not available

(hello-world-gtk:109667): Gtk-CRITICAL **: 14:50:30.759: Unable to connect to the accessibility
bus at 'unix:path=/run/user/1000/at-spi/bus_1,guid=65774ea9c81c70feb5c5007163669b8e': Could not
connect: Permission denied

Gdk-Message: 14:50:30.760: Failed to get file transfer portal: Could not connect: Permission
denied
```



Gtk Hello World - Extra Credit

Dani - do we want to talk about the errors and leave it as an exercise to the attendee to fix them? Or do you have thoughts on what to fix?

Example #2

Local deb

Staging local deb

What to expect:

- Learn how to build a snap from a local deb (could be a proprietary package or deb provided from somewhere else

extension-manager: Building the yaml for the deb

We have a deb binary from somewhere: gnome-shell-extension-manager

```
~/snap-examples$ mkdir -p extension-manager && cd extension-manager
~/snap-examples/extension-manager$ apt download gnome-shell-extension-manager
```

Using your favorite editor, create snapcraft.yaml:

```
~/snap-examples/extension-manager$ vim snapcraft.yaml
```

Ok what do we need? Same stuff as before :)

- metadata: name, version, summary, description, base, confinement
- apps: list of apps that will be built
- parts: the stuff that builds the apps

(Quick grab: https://github.com/hellsworth/hello-world-gtk)

extension-manager - Metadata

```
~/snap-examples/extension-manager$ vim snapcraft.yaml
```

```
name: extension-manager

version: '0.1'

summary: Utility for managing GNOME Shell Extensions

description: The GNOME shell extension manager is a tool for helping you manage your installed
GNOME shell extensions.

base: core22

confinement: strict
```

Recommended metadata: https://snapcraft.io/docs/adding-global-metadata

Complete list of metadata keys/values: https://snapcraft.io/docs/snapcraft-top-level-metadata

(Quick grab: https://github.com/hellsworth/extension-manager-snap)

extension-manager - Apps

```
~/snap-examples/extension-manager$ vim snapcraft.yaml


apps:
  extension-manager:
     command: usr/bin/extension-manager
     extensions: [gnome]
     plugs:
         - x11
         - wayland
```

(Quick grab: https://github.com/hellsworth/extension-manager-snap)

extension-manager - slots

```
~/snap-examples/extension-manager$ vim snapcraft.yaml


slots:
  extension-manager:
      interface: dbus
      bus: session
      name: com.mattjakeman.ExtensionManager
```

For a complete list of metadata keys/values: https://snapcraft.io/docs/adding-parts

(Quick grab: https://github.com/hellsworth/extension-manager-snap)

extension-manager - parts

```
~/snap-examples/extension-manager$ vim snapcraft.yaml

parts:
 extension-manager:
   plugin: dump
   source: gnome-shell-extension-manager_0.4.0-0ubuntu1_amd64.deb
   source-type: deb
   build-packages:
    - libtext-engine-dev
   stage-packages:
    - libtext-engine-dev
```

For a complete list of metadata keys/values: https://snapcraft.io/docs/adding-parts

(Quick grab: https://github.com/hellsworth/extension-manager-snap)

extension-manager - Build it!

Build command: “snapcraft” with options

- v = print stdout

- - debug = in case of failure, drop us into build environment

- - shell = in case of pass, drop us into build environment

```
~/snap-examples/extension-manager$ snapcraft -v --debug
Starting Snapcraft 7.2.1.post14+gitc7ce663f
Logging execution to '/home/heather/.cache/snapcraft/log/snapcraft-20221107-124402.435403.log'
  <snip>
Creating snap package...
Created snap package extension-manager_0.1_amd64.snap
```

(Quick grab: https://github.com/hellsworth/hello-world-gtk)

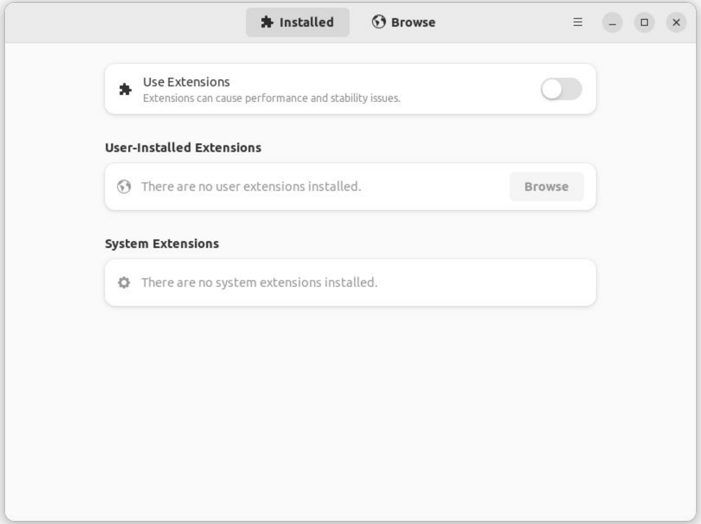
extension-manager - Install and Run it!

Build command: “snap” with options

- - dangerous = if not installed from store, circumvent store signature

```
~/snap-examples/extension-manager$ sudo snap install extension-manager_0.1_amd64.snap --dangerous

extension-manager 0.1 installed
```

```
$ snap run extension-manager
```



Use Extensions
Extensions can cause performance and stability issues.

User-Installed Extensions

There are no user extensions installed.

System Extensions

There are no system extensions installed.

Example #3

Stepping it up

gnome-calculator example

What to expect:

- Look at a more advanced example:

https://github.com/ubuntu/gnome-calculator.git

Key Concepts:

- build-snaps / cleanup
- override-build, override-prime
- meson plugin
- Remote source
- build and test

Your turn!
(let’s snap *your* app)

Thank you. Questions?