Creating gadget for Enlightenment

...and dig into the details

Raffaele Spinelli

November 23, 2019

Table of Contents

Table of Contents

What are gadgets

Gadgets are visible items (usually provided by a module) that can be placed by the user on the Shelf or on the Desktop
Gadgets allows to perform different tasks such as change the sound volume, monitor the battery status, access programs & iconified windows.

E gadgets classification

There are two kind of gadget one can build

- **Sandbox gadget** are standalone applications that Enlightenment can use as extensions to its desktop
- **Normal gadget** running inside of E

E gadgets classification

There are two kind of gadget one can build

- **Sandbox gadget** are standalone applications that Enlightenment can use as extensions to its desktop
- **Normal gadget** running inside of E

They are similar to modules in that they can be placed in containers, called gadget sites; they differ in that they are not internal to Enlightenment and do not run in the same process.

Sandbox gadget

Sandbox gadget traits

- Separate process
- Adhere to the ELM_MAIN loop
- Work also as a standalone application

Normal gadget

Normal traits

- Module based
- Module initialise the gadget

E gadgets documentation

Note

Although the Sandbox gadget are discouraged, the main documentation available on the website is only referring to this kind

E gadgets documentation

Note

Although the Sandbox gadget are discouraged, the main documentation available on the website is only referring to this kind

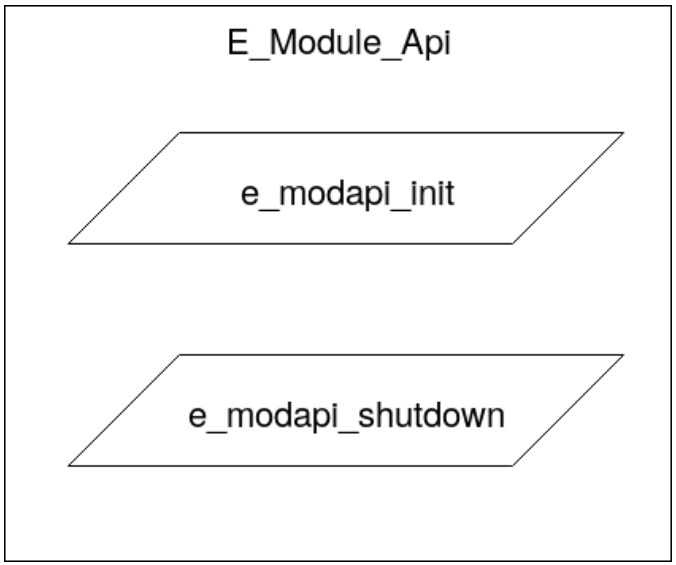
Remember

One should always refer to standard gadget and not to the sandbox gadget

Architecture outline

Keep in mind

Standard gadgets are written as modules.

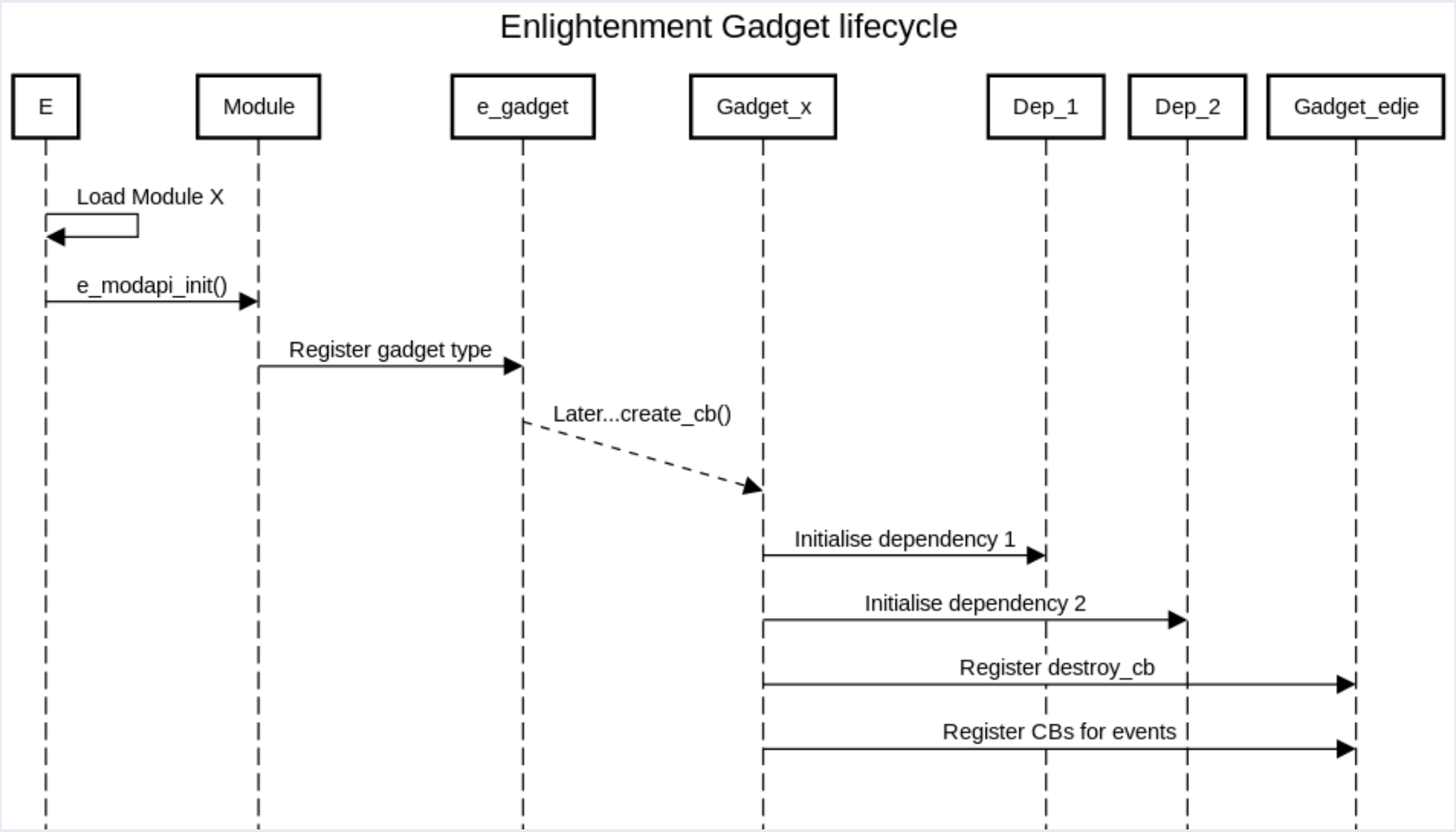


The initialization usually has three things to be done

- Initialise the logger
- Initialise the configuration and the menu entrie(s)
- Register the gadget e_gadget_type_add

Internal flow

Gadget creation



Internal flow

Gadget removal

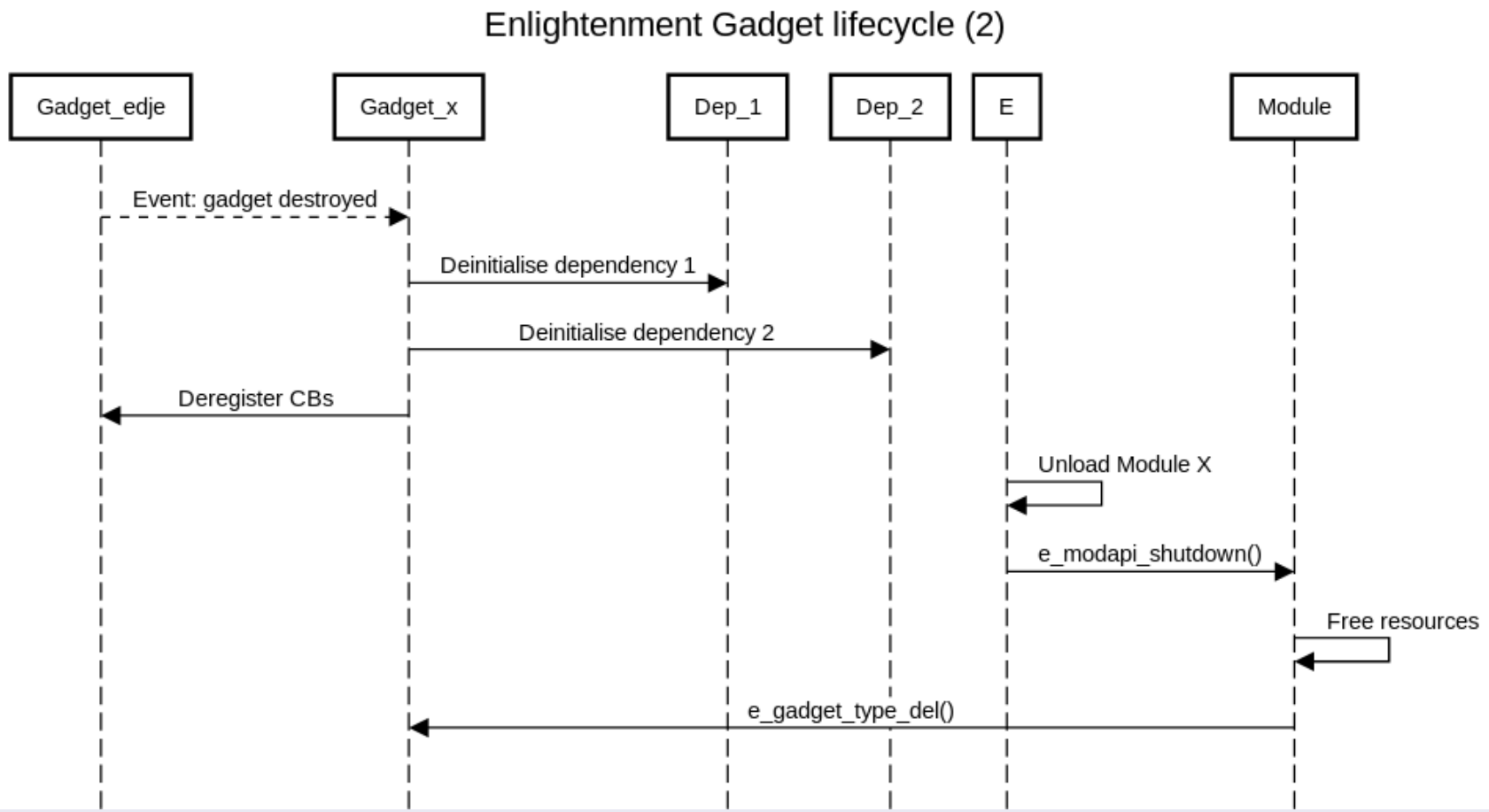
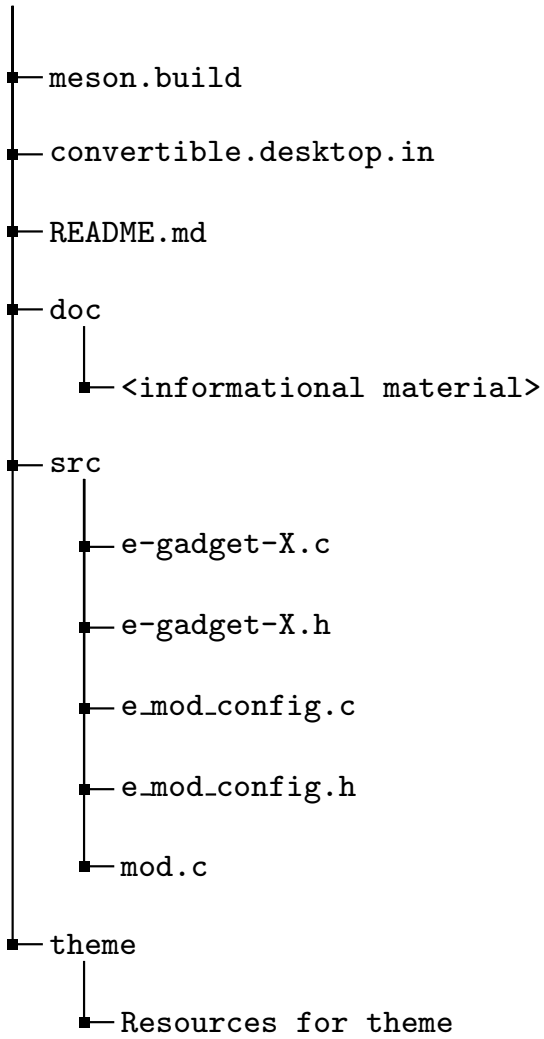


Table of Contents

Structure your project

Usually, you would like to have a folder for the theme of your gadget, one for the source code and you will need a desktop file and a medon.build file


meson.build
convertible.desktop.in
README.md
doc
<informational material>
src
e-gadget-X.c
e-gadget-X.h
e_mod_config.c
e_mod_config.h
mod.c
theme
Resources for theme

You may wish to make your gadget configurable.
That will imply:

- Have a schema of the settings information you are gonna handle

You may wish to make your gadget configurable.
That will imply:

- Have a schema of the settings information you are gonna handle
- Define a dialog box

You may wish to make your gadget configurable.
That will imply:

- Have a schema of the settings information you are gonna handle
- Define a dialog box
- Make sure the dialog box settings and the current config are correctly kept in sync

Data structure

```
static Convertible_Config *_config = NULL;
```

Data structure

```
static Convertible_Config *_config = NULL;
```

Handling config data

You need to implement the following functions

```
static void
_config_set(Convertible_Config *config);
static void*
 _create_data(E_Config_Dialog *cfg EINA_UNUSED);
static void
_free_data(E_Config_Dialog *c EINA_UNUSED,
E_Config_Dialog_Data *cf
);
```

Types

```
cd = E_CONFIG_DD_NEW("Convertible_Config",
                     Convertible_Config);
E_CONFIG_VAL(cd, Convertible_Config,
                     monitoring, UCHAR);
E_CONFIG_VAL(cd, Convertible_Config,
        disable_keyboard_on_rotation,
        UCHAR);
E_CONFIG_LIST(cd, Convertible_Config,
        rotatable_screen_configuration,
        c_zone);
```

Types

```
cd = E_CONFIG_DD_NEW("Convertible_Config",
                     Convertible_Config);
E_CONFIG_VAL(cd, Convertible_Config,
                     monitoring, UCHAR);
E_CONFIG_VAL(cd, Convertible_Config,
        disable_keyboard_on_rotation,
        UCHAR);
E_CONFIG_LIST(cd, Convertible_Config,
        rotatable_screen_configuration,
        c_zone);
```

UCHAR

Boolean must use UCHAR

Create a dialog box

```
v = E_NEW(E_Config_Dialog_View, 1);
v->create_cfdata = _create_data;
v->free_cfdata = _free_data;
v->basic.apply_cfdata = _basic_apply_data;
v->basic.create_widgets = _basic_create_widgets;

cfd = e_config_dialog_new(comp, "MyGadget␣Configuration", "E",
"windows/mygadget", NULL, 0, v, NULL);
```

- Customise logging by defining your own macros
- Initialise your logger
- Taking care of releasing the logger when the module is unloaded

- Perhaps in your e_gadget_logging.h file
- In **e_modapi_init**
- In **e_modapi_shutdown**

Define your logging functions

```
...
#undef CRIT
#undef ERR

extern int _convertible_log_dom;
#define CRIT(...)      EINA_LOG_DOM_CRIT(_convertible_log_dom, __VA_ARGS__)
#define ERR(...)       EINA_LOG_DOM_ERR(_convertible_log_dom, __VA_ARGS__)
```

Initialise logger

```
// Initialise the logger
_convertible_log_dom = eina_log_domain_register(
        "convertible",
        EINA_COLOR_LIGHTBLUE
);
```

De-initialise logger

```
eina_log_domain_unregister(_convertible_log_dom);
_convertible_log_dom = -1;
```

Table of Contents

Newbie questions

- How can I build the gadget?
- How can I test my gadget?
- How can I debug a gadget?

Newbie questions

- How can I build the gadget?
- How can I test my gadget?
- How can I debug a gadget?

Answers from doc + community

- Enlightenment has adopted Meson[a] as build system
- You can run a different session of Enlightenment under Xephyr
- You can use valgrind[b]

---

[a] `https://mesonbuild.com/`
[b] `http://valgrind.org/`

Meson build according to Wikipedia[1]

Meson

Meson is a software tool for automating the building (compiling) of software. Meson is free and open-source software written in Python, under the Apache License 2.0

Supports

Meson supports the C, C++, D, Fortran, Java, Rust and Vala languages,[5] and has a mechanism for handling dependencies called Wrap.
Meson supports GNU Compiler Collection, Clang, Microsoft Visual Studio and others.

[1] `https://en.wikipedia.org/wiki/Meson_(software)`

Important things to specify in the meson file

Meson file sections

- Dependecies
- Where to install the module
- Install additional files

Dependencies

```
dep_e = dependency('enlightenment')
dep_edje = dependency('edje' , required: true)
deps = [
        dep_e,
        dependency('elementary'),
        dependency('ecore'),
        dependency('eldbus')
]
```

Building files

```
build_files = [
'src/mod.c',
'src/dbus_acceleration.h',
'src/dbus_acceleration.c',
'src/e-gadget-convertible.h',
'src/e-gadget-convertible.c',
'src/accelerometer-orientation.h',
'src/convertible_logging.h',
'src/convertible.h',
'src/e_mod_config.h',
'src/e_mod_config.c'
]
```

Building files Output

```
inc = include_directories('.', '../../')

shared_module('module', build_files,
include_directories: inc,
name_prefix        : '',
dependencies       : [ dep_e, deps ],
install_dir        : join_paths([dir_gadgets, module_arch]),
install            : true,
link_args          : '-Wl,--unresolved-symbols=ignore-all'
)
```

Additional output

```
desktop_data = configuration_data()
desktop_data.set('GADGET_DIR', dir_gadgets)

configure_file(input: 'convertible.desktop.in',
output: 'module.desktop',
install: true,
install_dir: dir_gadgets,
configuration: desktop_data)

configure_file(input: 'theme/convertible.edj',
output: 'e-module-convertible.edj',
install: true,
install_dir: dir_gadgets,
copy: true)
```

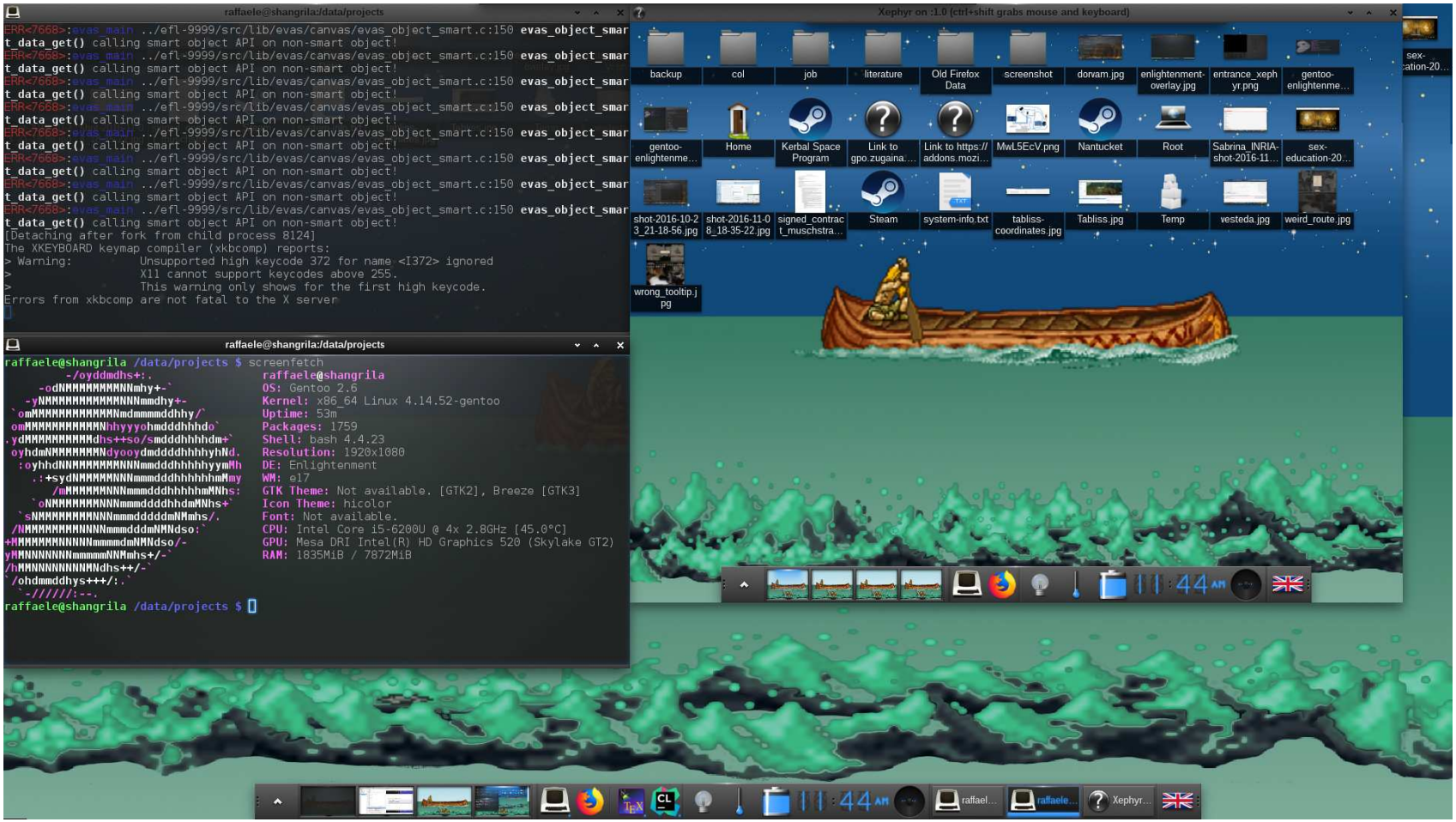
How to run your gadget

Load/unloading a module and creating/removing a gadget is tiring, but there are alternatives.

Xephyr session

In the enlightenment repository, you can find a great utility named xdebug.sh that will run another enlightenment process using Xephyr.

`https://git.enlightenment.org/core/enlightenment.git/tree/xdebug.sh`


raffaele@shangrila:/data/projects
ERR<7668>:evas_main ../efl-9999/src/lib/evas/canvas/evas_object_smart.c:150 evas_object_smart_data_get() calling smart object API on non-smart object!
[Detaching after fork from child process 8124]
The XKEYBOARD keymap compiler (xkbcomp) reports:
> Warning: Unsupported high keycode 372 for name <I372> ignored
> X11 cannot support keycodes above 255.
> This warning only shows for the first high keycode.
Errors from xkbcomp are not fatal to the X server
raffaele@shangrila:/data/projects
raffaele@shangrila /data/projects $ screenfetch
raffaele@shangrila
OS: Gentoo 2.6
Kernel: x86_64 Linux 4.14.52-gentoo
Uptime: 53m
Packages: 1759
Shell: bash 4.4.23
Resolution: 1920x1080
DE: Enlightenment
WM: e17
GTK Theme: Not available. [GTK2], Breeze [GTK3]
Icon Theme: hicolor
Font: Not available.
CPU: Intel Core i5-6200U @ 4x 2.8GHz [45.0°C]
GPU: Mesa DRI Intel(R) HD Graphics 520 (Skylake GT2)
RAM: 1835MiB / 7872MiB
raffaele@shangrila /data/projects $
Xephyr on :1.0 (ctrl+shift grabs mouse and keyboard)
backup
col
job
literature
Old Firefox Data
screenshot
dorvam.jpg
enlightenment-overlay.jpg
entrance_xephyr.png
gentoo-enlightenme...
Home
Kerbal Space Program
Link to gpo.zugaina....
Link to https://addons.mozi...
MwL5EcV.png
Nantucket
Root
Sabrina_INRIA-shot-2016-11...
sex-education-20...
shot-2016-10-23_21-18-56.jpg
shot-2016-11-08_18-35-22.jpg
signed_contract_muschstra...
Steam
system-info.txt
tabliss-coordinates.jpg
Tabliss.jpg
Temp
vesteda.jpg
weird_route.jpg
wrong_tooltip.jpg
44 AM

Gadget for automatic screen rotation according to gyroscope data

- Uses eldbus to read Gyroscope data through IIO-Sensor-Proxy[2]
- Project: `https://github.com/rafspiny/convertible/`
- Story: `https://www.rafspiny.eu/blog/enlightenment-gadget-for-laptop-auto-rotate-function`

[2] `https://github.com/hadess/iio-sensor-proxy`

Useful links

- Enlightenment Main Website
- C API documentation
- A Software engineering platform
- The source of truth AKA GIT
- Eldbus

Thanks

Comments?