Writing Really Rad GTK+ & GNOME Applications in C, Python or Java

Andrew Cowie
Operational Dynamics

Davyd Madeley
Fugro Seismic Imaging

Who Are We?

Andrew Cowie

spends an awful lot of time programming for someone who is actually a suit. He started with C in the early 80s, picked up Java in 1997, and now, 10 years later, is the maintainer of the `java-gnome` project.

Davyd Madeley

is a professional software engineer and electronic engineering student. By night he is the `gnome-applets` maintainer and a contributor to GNOME. He plays the tenor saxophone.

An Overview

- Why choose GTK+ for your application?
- GTK+ Fundamentals
  - Building a UI
  - Box packing
  - The main loop & signals
- Getting started (in C)
- Window tricks (in Java)
- Complex data models (in Python)

Why Would You Choose GTK+?

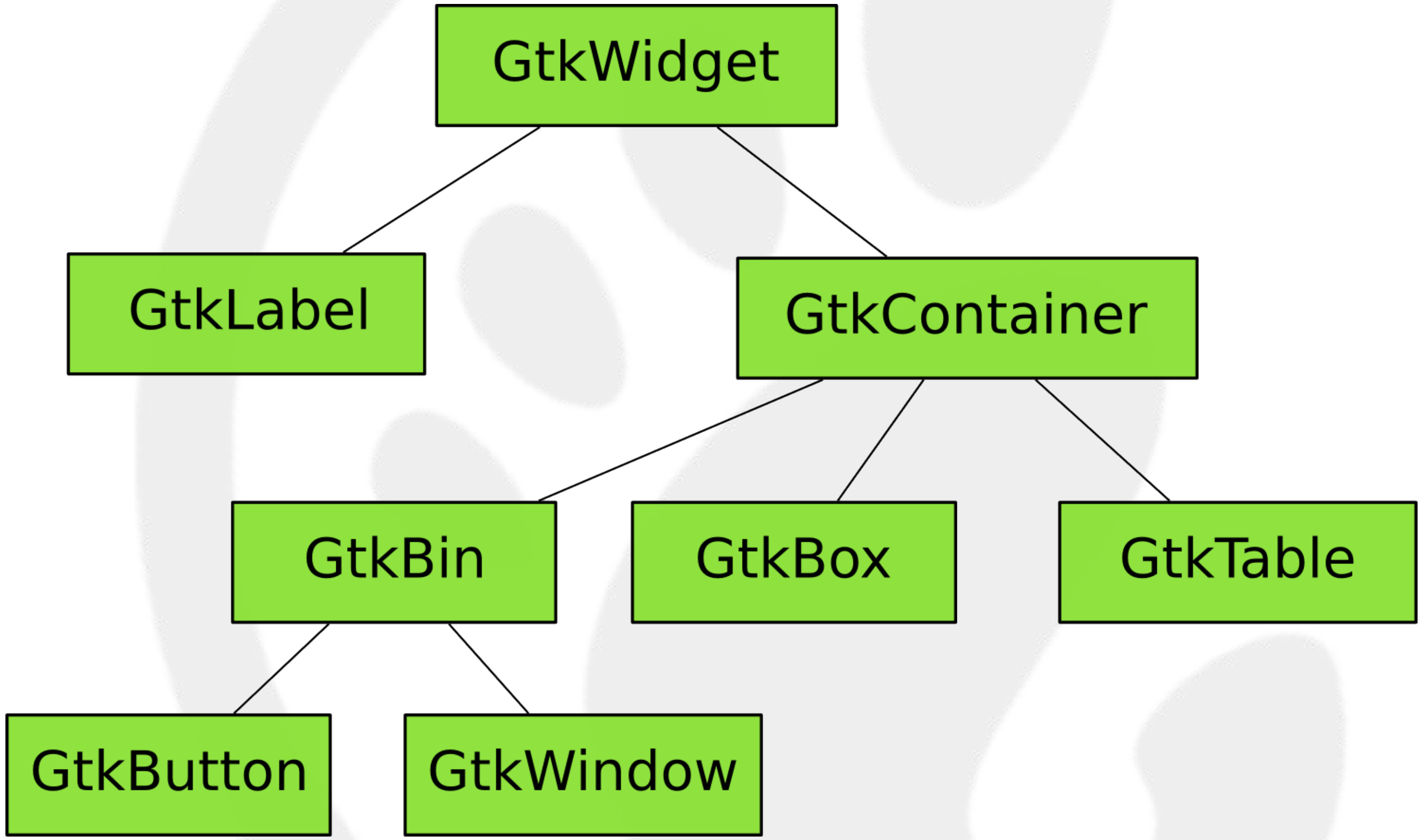
- Fast, flexible, ubiquitous
- Multi-platform
  - **Linux**, Unix, Mac OS, Win32, and more
- Many languages
  - **C**, **Python** and **Java**
  - Perl, C++, Ruby, Haskell, C#, PHP, OCml, Eiffel, Erlang, Guile/Scheme/Lisp, Lua, Octave, D, TCL, Smalltalk, and more!
- LGPL

A Word on Versions

- Today we're using the following:
  - gcc 4.1.x
  - GTK+ 2.10.x
  - Python 2.4
  - pyGTK 2.10
  - Sun Java 1.5 (& Free Java too!)
  - Eclipse 3.2.x
  - java-gnome 4.0
  - Glade 3.1.x

Widgets 'n stuff

- all displayed items are a `GtkWidget`; all interfaces are built down from a “top level”, inevitably `GtkWindow`



Building a UI

- You can write code …
  - Programmatically create elaborate custom content, dynamic layouts, and smaller Widgets

C Demo!

A `GtkWindow` with a `GtkButton` in it!

Compiling

```
gcc -o demo \
`pkg-config --cflags --libs \
 gtk+-2.0` demo.c
```

Building a UI

- You can write code ...
  - Programmatically create elaborate custom content, dynamic layouts, and smaller Widgets
- or use Glade ...
  - Great for big, complex windows with lots of Layout

C Demo!

A `GtkWindow` with a `GtkButton` with Glade!

Building a UI

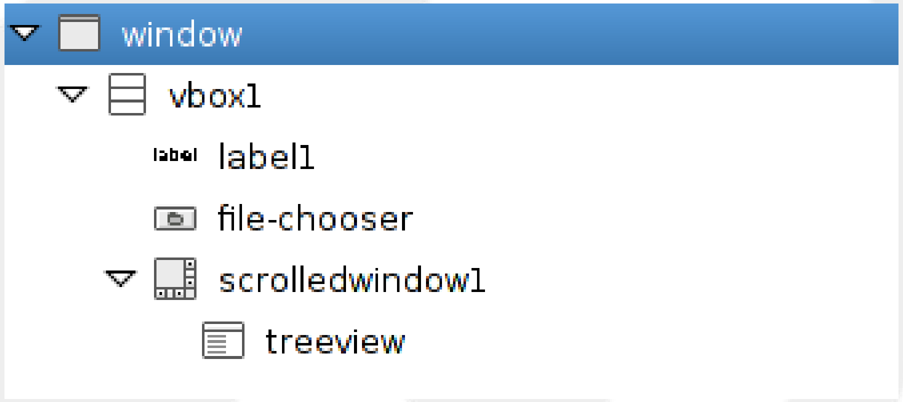
- You can write code ...
  - Programmatically create elaborate custom content, dynamic layouts, and smaller Widgets
- or use Glade ...
  - Great for big, complex windows with lots of Layout
- or do both simultaneously!
  - no point using Glade if coding it directly is less lines of code
  - Use Glade for most of Window (ie, Labels) and code for the dynamically generated bits

Box Packing

GTK+ uses a “box packing” model.

Box Packing

- Start a `GtkWindow`
- Pack a `GtkVBox` into the Window
- Pack a `GtkLabel` into the VBox
- Pack a `GtkScrolledWindow` into the VBox
- Pack a `GtkTreeView` into the ScrolledWindow


```
▽ window
    ▽ vbox1
        label1
        file-chooser
        ▽ scrolledwindow1
            treeview
```

Glade Demo!

Using
Glade
to do complex
Box packing
layouts

The Main Loop

- GUI programming is ***event driven*** programming
- The main loop polls *sources* for events
- events include user activity (keyboard or mouse), I/O, or a timeout
- events issued as named signals; register callbacks for signals you want to react to

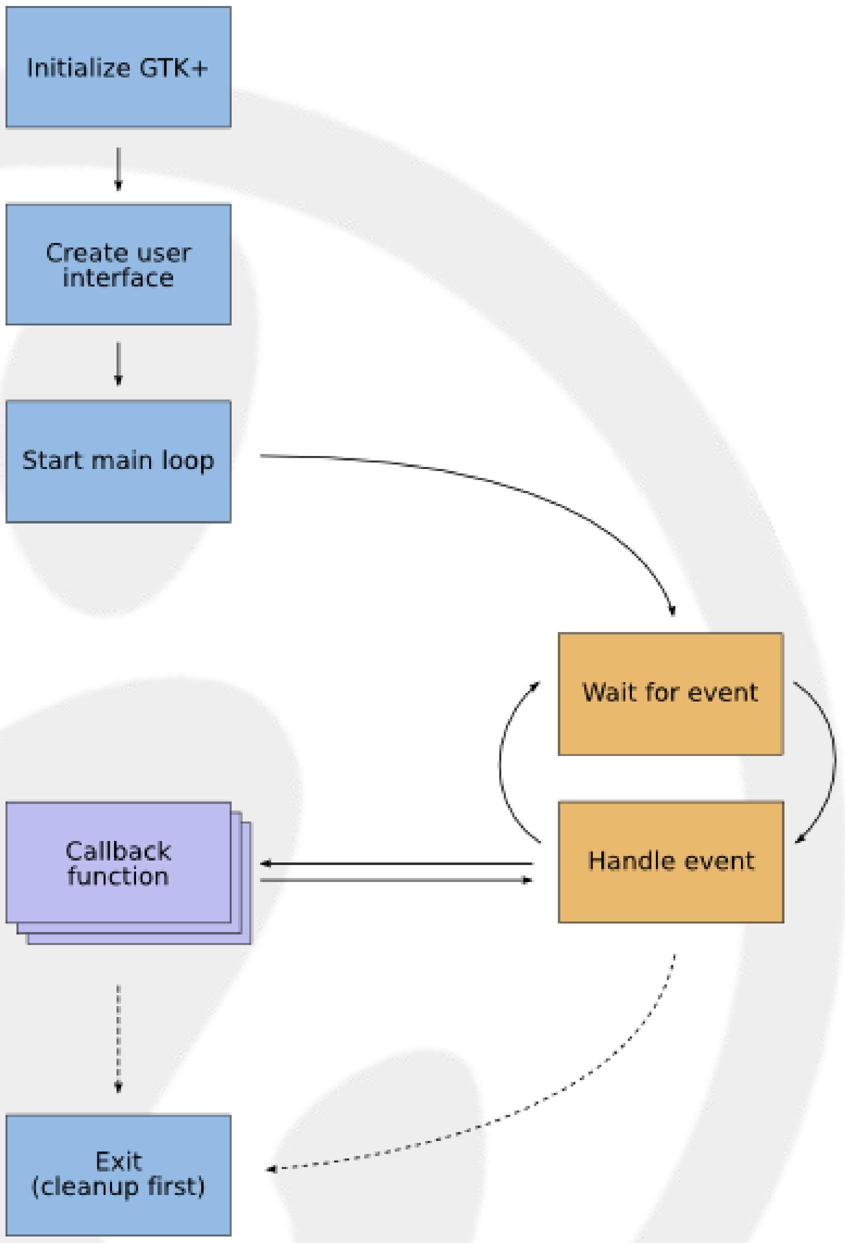
The Main Loop

Callbacks for events are issued from the main loop...

... one at a time

... and it's single threaded!

DON'T BLOCK THE MAIN LOOP!



Signals

- Signals are connected to `GObject`s
- Often you pass 4 things:
  - object
  - signal name
  - callback function
  - optional free-form “user data”
- Prototype for each callback in API docs
- Some callbacks return information to GTK+ (eg a `gboolean`)

Signals – C

```
g_signal_connect(my_gobject,
                 “notify::parent”,
                 G_CALLBACK(notify_parent_cb),
                 NULL);



void notify_parent_cb(GObject *my_gobject,
                      GParamSpec arg1,
                      gpointer user_data)
{

   ...

}
```

C Demo!

Hooking up a signal

Signals

- Some signals already have handlers registered
  - eg. expose-event
- Some signals are passed up the widget tree from your widget all the way to the toplevel
  - eg. expose-event, enter-notify-event
  - You can choose whether or not to stop these in your signal handler by returning True or False

Java Demo!

Same code,
different language:
Java

delete-event

Closing a Window

!=

Terminating application

Beware the main loop!

GtkFileChooser

Choose a file,

any file

Python Demo!

Same code,
different language:

Python

GtkTreeView

- Can display trees or lists of data
- Uses an model, view, control (MVC) paradigm
- You need three things:
  - a `GtkTreeView`
  - a `GtkTreeModel` (`GtkTreeStore`, `GtkListStore` or write your own)
  - `GtkCellRenderer`s
- You can store more data in a row than you display (handy!)

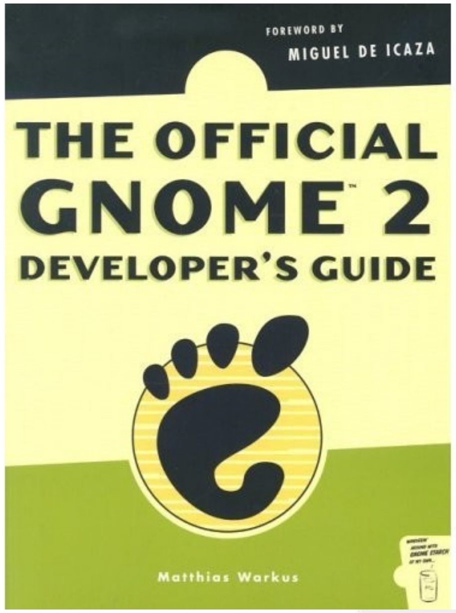
Python Demo!

See the `gtk.TreeView` for the Forrest

Getting More Out of GTK+/GNOME

- GConf – store configuration data
- GNOME-VFS – access data over networks
- Cairo – antialiased vector graphics
- GooCanvas – Cairo based canvas widget
- D-BUS – cross-desktop IPC with GLib tie-in
- Soup – HTTP, XML-RPC and SOAP libraries
- libwnck – Access window information
- libnotify – Popup balloons

Would Ye Like To Know More?



- In C:
  - http://www.gtk.org/tutorial/
  - Matthias Warkus, *The Official GNOME 2 Developer's Guide* (No Starch Press, 2004)
  - Devhelp
- In Java:
  - http://java-gnome.sourceforge.net/4.0/doc/
- In Python:
  - http://www.pygtk.org/pygtk2tutorial/index.html

Fin ;)

Questions?

www.davyd.id.au/articles.shtml

operationaldynamics.com/talks