D ♥ GTK

Gerald Nunn
http://www.gexperts.com
It's ugly, but it gets you there.
Bias Check

- Started with Borland’s Delphi building Desktop Apps in 90s
- Switched to Java and Java Enterprise Edition in 2000
- Some experience with Python for IT automation
- Work for Red Hat as a Middleware Solution Architect
  - No formal involvement with GNOME or GTK
- Biases
  - Prefer strongly and statically typed languages
  - Believe tooling is important
  - Spaces not Tabs :)
- Started using D in late 2015
GTK+
What is GTK+?

- A cross-platform GUI Toolkit for Linux, Windows and OSX
- One of the two primary GUI toolkits for Linux
- The toolkit used by the GNOME Desktop Environment
- Primarily written in C with multiple language bindings
  - JavaScript
  - Python
  - C++
  - D
  - Rust
  - Vala
  - Pascal
  - Haskell
  - Go (partial)
  - etc...
GTK+, more than you ever wanted to know

- GTK+ is based around the GLib Object System
  - Aka GObject
- It is reference counted
- GTK+ 3 is current stable version, GTK+ 4 in development
- It uses an object oriented C API
  - Object state is represented by a class specific struct
  - Methods follow convention of `{prefix}_{class}_{method}`
    - `gtk_widget_show`
    - `gtk_container_add`
    - `g_settings_set_boolean`
  - Relies heavily on inheritance
- GIR files available in XML containing GOBject introspection information
Writing a GTK+ App

The Usual Suspects…

… from Middle-Earth
JavaScript

- Gjs is the JavaScript binding for GNOME
- Based on Mozilla SpiderMonkey
- Used extensively in GNOME Shell

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Large eco-system of tools and libraries</td>
<td>Dynamically and weakly typed language</td>
</tr>
<tr>
<td>Easy to learn</td>
<td>Gjs Not widely used outside of GNOME (Shell, Polari, Maps)</td>
</tr>
<tr>
<td>Memory usage on par with Python</td>
<td>Slow performance</td>
</tr>
<tr>
<td></td>
<td>Sub-optimal language design</td>
</tr>
</tbody>
</table>
Python

- Very popular choice for writing GTK applications (Lollypop, Terminator, etc..)
- PyGObject provides the GTK language bindings

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Large eco-system of tools and libraries</td>
<td>Dynamically typed language</td>
</tr>
<tr>
<td>Widely used in GTK applications</td>
<td>Reasonably performant but not natively compiled</td>
</tr>
<tr>
<td>Highly productive and easy to learn</td>
<td>Whitespace as a delimiter</td>
</tr>
<tr>
<td>First class support for creating GObjects</td>
<td></td>
</tr>
</tbody>
</table>
C

- The lingua franca of GTK
- No bindings required

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Large eco-system of tools and libraries</td>
<td>Not as productive as alternatives</td>
</tr>
<tr>
<td>Very widely used in GTK applications</td>
<td>High learning curve</td>
</tr>
<tr>
<td>Native compilation, excellent performance</td>
<td>Inconsistent memory management</td>
</tr>
<tr>
<td></td>
<td>Unsafe</td>
</tr>
</tbody>
</table>
C++

- C but better..maybe..
- Bindings provided by gtkmm

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Large eco-system of tools and libraries</td>
<td>Not as productive as alternatives</td>
</tr>
<tr>
<td>Used in a number of GTK applications</td>
<td>Very high learning curve</td>
</tr>
<tr>
<td>Excellent performance</td>
<td>Very Slow compilation</td>
</tr>
<tr>
<td>First class support for creating GObjects</td>
<td>Safer than C but not perfect</td>
</tr>
</tbody>
</table>
Vala

- Language designed explicitly to support GObject and GTK+ development
- Default language for Elementary OS

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Easy to learn for people coming from C based languages</td>
<td>Not a general purpose language, single focus</td>
</tr>
<tr>
<td>Natively compiled, fast performance</td>
<td>Limited tool support</td>
</tr>
<tr>
<td>Memory management via RC</td>
<td>Cross-compilation to C can make debugging challenging</td>
</tr>
<tr>
<td>Highly productive with many abstractions available</td>
<td></td>
</tr>
</tbody>
</table>
Writing a GTK App

The Up-and-Comers
Rust

- New language from Mozilla
- Built on concept of memory safety
- GTK-RS is the binding library

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Natively compiled, fast performance</td>
<td>High learning curve, borrow-checker semantics</td>
</tr>
<tr>
<td>Memory management and safety</td>
<td>Decent but not great tooling</td>
</tr>
<tr>
<td>Widely hyped language with rapidly growing base of tools</td>
<td>Few GTK applications using language</td>
</tr>
<tr>
<td>GNOME developers actively engaged</td>
<td>Slow compilation times</td>
</tr>
<tr>
<td>Excels at libraries and low-level use cases</td>
<td></td>
</tr>
</tbody>
</table>
You already know the story...

Bindings provided by GtkD

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Natively compiled, fast performance</td>
<td>Few GTK applications using language</td>
</tr>
<tr>
<td>Memory management via GC</td>
<td>Multiple compilers and ABIs</td>
</tr>
<tr>
<td>Fast compile times</td>
<td>Decent but not great tooling</td>
</tr>
<tr>
<td>Easy to learn particularly when coming from C</td>
<td></td>
</tr>
<tr>
<td>family languages</td>
<td></td>
</tr>
</tbody>
</table>
GTK+ and D

The Argument for D
The Original Vala Use Case

The Sweet Spot...
However, Vala Has No Future (IMHO)

- Few developers
- Little development
- No use outside of GTK+ Development
My Ideal Future...

The Sweet Spot...
Why D for GTK Applications?

- **GtkD advantages**
  - Full coverage of GTK+ API plus more
  - Generated API based on GObject Introspection
- High performance, fast compilation
- Easy interfacing with C if you need to go low-level
- CTFE potential as C Macros replacement
- GC is productive for GUIs
- Maintenance benefits of static typing
- Easy transition for Vala developers
Why Not Rust?

✗ High learning curve compared to D
✗ Memory Safety is cool...
  o Is the benefit worth the cost?
  o Looks phenomenal when compared to C…
  o Less so compared to other modern languages
  o Dependencies on unsafe code

✓ Great for Libraries and Low-Level Work
Success Story
Tilix Experience Report
Tilix

```
"debugVersions": ["Destructors"
,
  "name": "dynamic",
  "targetType": "executable",
  "libs": ["gtk-3.0"],
  "libs-linux": ["X11"],
  "lflags": ["-defaultlib=gtk-3.so"],
  "versions": ["StdStringLoggerDisableTrace"]
]
```

```
[gunn@gunn-xps15 ~]$ ls
Desktop  Downloads  Music  Templates
Development  GoogleDrive  Pictures  Videos
Documents  GoogleDriveRedHat  Public
[gunn@gunn-xps15 ~]$ 
```
Tilix - Fast Facts

- Started in November 2015
- Written in D using the GtkD bindings
- Supports GTK+ 3.14 to 3.24
- Complies with GNOME HIG
- Is my second GTK+ program
  - Visual Grep was first, learning experience
Tilix Successes

- Packages available for every major Linux Distro
- Official packages for Debian
- Official package for Fedora will be available in Fedora 26
- Default Terminal for Ubuntu Budgie flavor
- Localized in ~25 languages
- Github:
  - 74 Contributors
  - 54 Releases
  - 185 PRs
  - ~700 issues, ~670 closed
  - ~1300 stars, 72 forks
  - 2nd most starred D Application on GitHub
Lessons Learned - The Good Stuff

- GtkD bindings are mature
- Linux packaging is hard, leave it to the experts
- Distro specific bugs are unfortunately common…
- … so be restrictive on what you support
- GNU gettext works great using Vala as input language
Lessons Learned - Regrets

- Should have made more effort up front to leverage unique D features
- Check trademarks before choosing project name
- Take the high road when interacting with users
A tiling terminal emulator for Linux using GTK+ 3  
https://gnunn1.github.io/tilix-web

https://github.com/gnunn1/tilix
A Call To Action
What You Can Do

● Write GTK+ Applications
  ○ More Apps = More Recognition
  ○ D & GTK Template App from GNOME Developer Carlos Soriano
    ■ https://gitlab.com/csoriano/GtkDApp

● Contribute to GtkD Development
  ○ Create higher level API abstractions

● Contribute To D Development

● Improve Linux packaging for D Applications
  ○ https://gist.github.com/ximion/fe6264481319dd94c8308b1ea4e8207a
Questions?
Desktop Applications

Yes Matilda, Desktop Applications Are Still Relevant...
Desktop Applications - Still Relevant?

“There are three consumer devices that are leading tablets by a large margin: TVs, smartphones, and computers. It seems unlikely that the tablet will ever displace these devices.”

“What about tablets? People say they’re just as good a PC for making content.

Sure they are, once you connect a keyboard and a mouse to them. Then guess what? That’s a PC too!”

“Well, it boils down to the reality that almost every piece of technology you use and much of the media content you consume, is designed and built on PCs.”

http://www.telegraph.co.uk/technology/2017/01/11/tablet-sales-fall-third-successive-year-pc-market-stabilises/amp/
Desktop Applications

Down but not out....

Five Years Past Peak PC
Worldwide PC shipments from 2006 to 2016 (in million units)
Desktops Are Evolving

Convergence?

It's a desktop! It's a tablet!

The PC is evolving, not disappearing, analysts say

Content Creators Need More Power
Agenda

- Bias Check
- Desktop Applications, Not Dead Yet...
- What is GTK?
- Writing a GTK App
  - The Usual Suspects
  - The Up-and-Comers
- D and GTK: The Perfect Match
- Experience Report: Tilix
<table>
<thead>
<tr>
<th>Commits</th>
<th>Almost 2000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contributors</td>
<td>74</td>
</tr>
<tr>
<td>Releases</td>
<td>54</td>
</tr>
<tr>
<td>Issues</td>
<td>~ 700 (~30 open)</td>
</tr>
<tr>
<td>Starred</td>
<td>1316</td>
</tr>
</tbody>
</table>

Second most starred D language project on Github: [https://github.com/search?l=D&q=stars%3A%3E1&s=stars&type=Repositories](https://github.com/search?l=D&q=stars%3A%3E1&s=stars&type=Repositories)
Desktop Market Evolving

Software for Consumers

Software for Professionals