Enlightenment Developer Day

2016, OpenWide Offices, Paris

Carsten Haitzler
<raster@rasterman.com>
<c.haitzler@samsung.com>
E* Stuff

What on earth are we doing here?
2 Major Areas of Focus

- **Enlightenment**
  - Desktop/Mobile/.. “Environment”
  - Apps
  - Services

- **EFL**
  - APIs
  - Services (related to APIs)
  - Developer Tools
2 Major Areas of Focus

- Enlightenment
  - Desktop/Mobile/.. “Environment”
  - Apps
  - Services

- EFL
  - APIs
  - Services (related to APIs)
  - Developer Tools
Currently working on EFL Interfaces

- EO object system + Interface mechanisms stabilized
  - Early July release of 1.18 with this
  - Not everything in EFL will be “EOified” by 1.18
    - 1.19, 1.20 ... will add more
- Goal should be:
  - “EOify” everything public API should expose
    - Exception - Eina (basic data structs and system API)
  - Everything else internal

Marching on to EFL 2.0

- EO Interfaces will be EFL 2.0
• Needing major overhaul
  – Ecore Con
  – Ecore IPC
  – Efreet
  – Ethumb
  – Eeze
  – Ecore Audio
  – Ecore File / Eina File / EIO

• Definitely have public API
  – Ecore
  – Edje
  – Efl
  – Eina
  – Eldbus
  – Elementary
  – Eo
  – Evas
  – Emotion
EFL

- Maybe expose by wrappers
  - Eet
  - Emile
  - Ephysics
  - Ecore Avahi

- The rest should be internal
  - Adjust source tree to show this

- Need to fix build system
  - Not useful for development
  - Way too slow for small changes

- Find legacy code and isolate it
EFL 2.0

- Once we have everything cleaned up
  - And we’ve wrapped/exposed via EO all APIs we want to
  - And we’ve cleaned up/redone the previous libs
  - And we’ve cleaned up our tree and build
  - And we’ve isolated legacy API code
  - ...
- Look at EFL 2.0
  - We may fix/clean/improve some EO/EFL Interfaces for 2.0 too
EFL Services

- Current Services
  - Efreetd (needs love/redesign)
  - Ethumbd (needs redesign or removal)

- Services we may need in future
  - Download manager + cache (download API and vpath integration)
  - App Launcher
    - Split off from Enlightenment doing it itself
    - Handle sandboxing/security etc.
    - Perhaps part of Efreet/replacement?
- ... ?
EFL Tools

- Tools
  - Debug
  - IDE
  - GUI Builders
  - Edje Editing
  - Project creation
  - Project upload/download/install

Clouseau, ... ?
EDI
Erigo
Enventor, Eflete, ... ?
EDI, ... ?
... Marrakesh (waiting on http server)
We need more portability abstractions

- Our native world is Linux
- Windows requires a lot
- OSX requires some
- Even Tizen creates portability issues
  - People have to PORT EFL Apps to Tizen - “CRAZY”
    - A Lot of the fault lies in Tizen’s ballcourt due to not being “normal” Linux
- Haven’t even looked at Android, iOS
Portability issues

- Need to hide/abstract main + argv/argc
  - Because Tizen is not normal Linux
  - Because Android is the same (and Tizen just mimics Android whenever it can)
  - ...
  - Means we also abstract/handle instance sharing
    - Terminology 1 instance support
- Need to pass in EFL version app is compiled against to EFL
  - To be able to mimic bug compatibility for older EFL versions in newer EFL releases
Efreet

- Totally tied to XDG - needs abstraction
  - To work on Windows, Tizen etc.
  - Find other apps by name/id and launch
  - Allow for indirect launching (not direct fork + exec)
  - Find icons for apps
  - Find apps for given mime types
- Probably should use native local EFL data store + importers/syncers to OS
Eeze

- Need a real device abstraction that does not exposes sysfs
- Likely needs special case support device type by type
Elementary Fileselector

- Needs to abstract OS features
  - Removable Devices
  - “Standard” Locations like “My Documents” / “~/Documents” / …
- Needs to be a lot more like Enlightenment Fileselector
  - Proper icons, good incremental list, delete/make dir/rename/file info
- Needs abstractions for sandboxing
  - Need to allow it to be out-of-process with fd-passing to requester
Widgets

- Needs love
  - Clean code up
  - Refactor to share more with inheritance etc.
  - Theme work needs to be slick and professional
    - No more “programmer art” level quality
- Bugs fixed
QA

- Tests
  - Need lots more
  - Tests have to be testing correct behavior, not invalid inputs FIRST
Tools

- Enventor & Eflete doing well...
- But now is time to focus on the programming - need and IDE - EDI
  - Make it eo aware
  - Integrate with higher level knowledge
    - int r, int g, int b, int a == color
    - const char *file == vpath capable reference (display file chooser button/preview)
    - ...
  - Once C is good, improve C++ support
  - Look at RAD language support (Lua, JS, Python)
Tools

- Ensure we also solve
  - Project creation (one-click create build tree with data files and sample source)
  - Building (build file maintenance - maybe just use Marrakesh?)
  - Packaging (Marrakesh + generators for RPM spec, deb pkg files, PKGBUILD, ...?)
  - Distribution (Marrakesh?)