1) Modern Controllers
2) Why a library
3) Users and Mappings
4) Demo!
5) Bonus Stuff...
Modern Controllers
Modern Controllers

"Next Gen" Controllers!

- USB HID devices
- "HD" Screens
- Huge functionality!
- Application support...
Modern Controllers

MIDI Mapping!

- APC40 released in 2014
  - USB MIDI device
  - Can't do full updates
Modern Controllers

HD Screens?

- MIDI messages
  - SysEx?
  - No thanks :)

No thanks :(
Modern Controllers

Need for a better API

- Software ⇔ Hardware
  - Easier communication
  - Faster to support modern features
- Required for Tight Integration
  - Input Multiplexing
Why A Library?
A Library for Controller Access

Developer implements library support

- Growing list of devices supported
- New devices added are "free"
- Don't re-implement device support
  - Testing is a time-sink!
Generic Event API

Abstract away the Device

- Input from Events
  - Button, Slider, Encoder, Grid
- LED feedback function
- API to blit pixels to a screen
Linux Support

Community Supported Devices

- Once-off implementations of little value
  - (Unless user has that exact controller)
- Centralize support in a library
  - Enabled by "Generic Events"
Fancy Features

Hotplug

- Essential to save a Musician on stage
- Difficult to implement
  - Time consuming to test
Controllers

MIDI
USB HID
BLULETOOTH
OSC
ARDUINO SERIAL

Software

DAW SOFTWARE
SEQUENCER
AUDIO EFFECTS RACK
LV2 PLUGINS
LIVE LOOPERS

Ctlra Library
Any Controller to Any Software!
Users and Mappings
Mappings and User-eXperience

Simple Workflow
- Can use software provided mapping
  - Just like MIDI mapping
- Not as powerful, but achieves goal
  - Mapping exists? Use it!

Power User
- Can "script" controls as required
- Has huge flexibility and power
- Creates awesome mappings
Power User

Uniting Software and Hardware

- Match functionality of controller to software
  - Just as the user requires
  - Depends on Hardware / Software combo
  - Depends on Musicians Workflow
Timeline (Just for fun :)

Bought a USB HID drum-pad controller

Jan '16

Integrated drum-pad and Fabla 2.0

Spring '16

Demo Fabla 2.0 with a specific USB HID device

miniLAC '16

Understand requirements and design v1 "ctlr" API

Summer '16

Implement v1 and discuss with other developers, design v2

Autumn '16
Development of Ctlra codebase

Finish initial implementation
Write Ctlra paper for LAC

Mixxx integration and testing of v2 Ctlra API
(Specifically Hotplug)

Minor updates to API,
Build out demo apps

Ctlra library release!
Demos!
Demo

1. Simple Events
2. Simple Feedback
3. Tight Integration
4. Scripting in C
5. DJ Hotplug
Next Steps

1. Discuss Event Loops
2. Integrate with LV2 Atoms
3. Get Applications using it
4. Discuss "sharing" of devices
5. Support for BlueTooth, MIDI, OSC, Arduino, Serial etc.
Questions
Bonus Demo

AVTK + Ctrlra
Bonus Slides!

Woop Woop :)
Event Loops

How to manage threading / event handling

- Current API has `idle()` function
  - Must be called periodically
- Expose events via ringbuffer?
  - How to handle hotplug of new device.. New ring?
LV2 Atom Integration

Ctrlra Events map to LV2 Atoms quite nicely...

- Could the host pass these events "through" to the plugin?
  - Abstract the environment from the plugin?
  - Enable "sharing" of a device?
- Requires "Options" extension function to accept new device?
USB Device Access

LibUSB

- Enables bulk, interrupt, isochronous, control endpoints
- API provides Sync and ASync modes
  - Async required for latency with multiple devices

HID Raw

- Won't support all devices (Screens are often USB "bulk" endpoints)
- Works better for certain devices..?
Arrival time of Event?

- Is it required at library level?
  - Time, Threading and ASync events... Fun!
- Expose an application defined callback to "set" the timestamp?