Chorus

Stephen Wu
Interactive Data Systems Group
Summer 2017

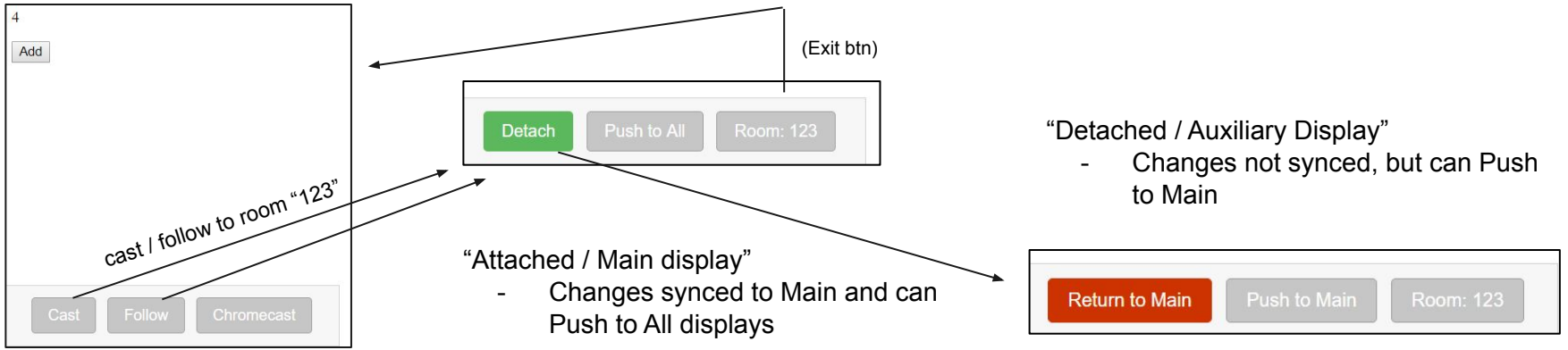
Overview

- **Multi-device data coordination for JS-based data visualizations**
  - Essentially “chatrooms” for data
- 5 demos (in order of creation)
  - TBDBITL pie chart with custom events (D3.js)
  - Collaborative keyboard and live bar chart (MIDI.js)
  - Airplane flight dashboard (Crossfilter.js)
  - Collaborative map (Leaflet.js)
  - Crossroads Integration (React, Leaflet.js, etc.)

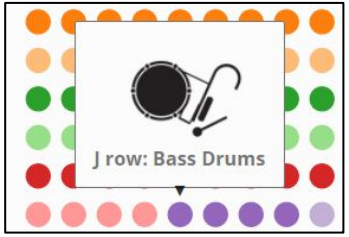
Chorus toolbar

States: “Dormant” —> “Attached, Main” —> “Detached, Auxiliary”
- Testing done with basic “Add” button demo.

Rules: (1) Keep data store in global _data var, (2) Run function chorusUpdate() when data is updated,
(3) Create global function chorusRender() that eventually renders data correctly, (4) Set chorusChromecast to true/false



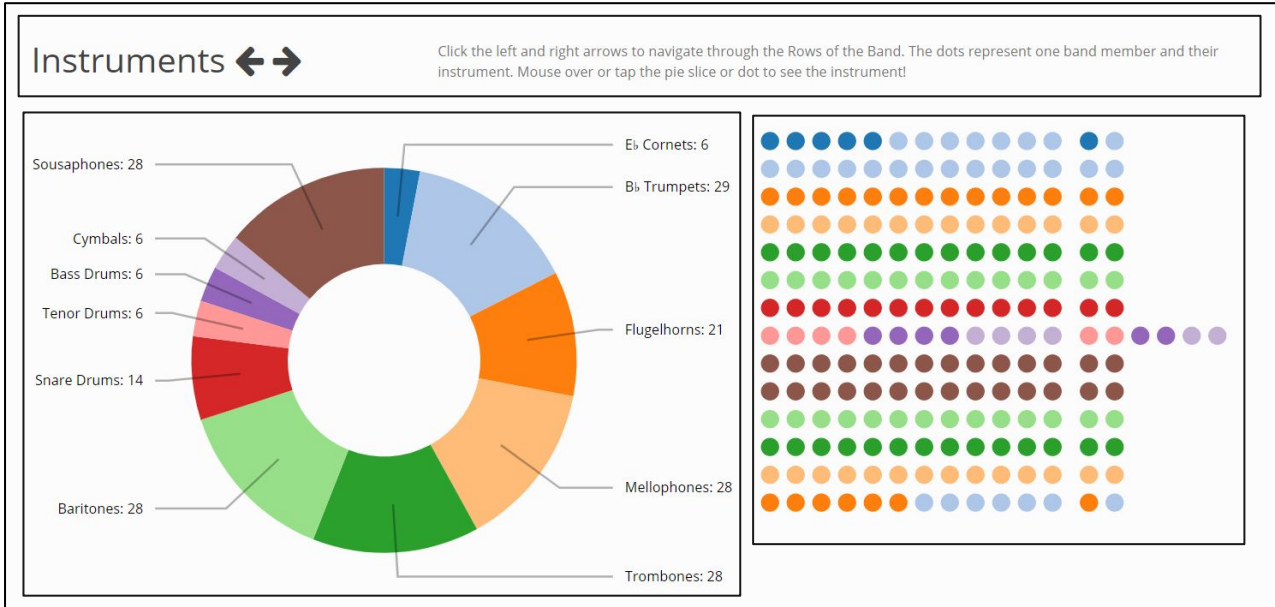
TBDBITL pie chart with custom events



3 components

- **Arrows that fire custom events** (Next Row / Previous Row)
- **Pie chart** showing breakdown of instrumentation (D3)
- **Dot chart** showing each individual line of members (D3)
- \+ Tooltips for each dot / pie portion showing instrument

Each were separated in 3 windows, synced with little-to-no latency over Socket.io. **Stored data were the current instruments in the pie & dot charts.**



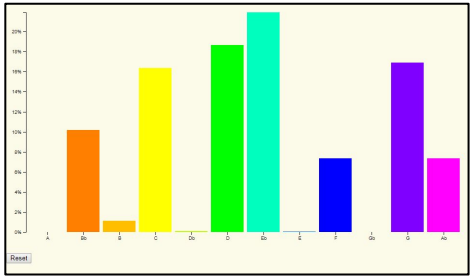
Collaborative keyboard and live bar chart

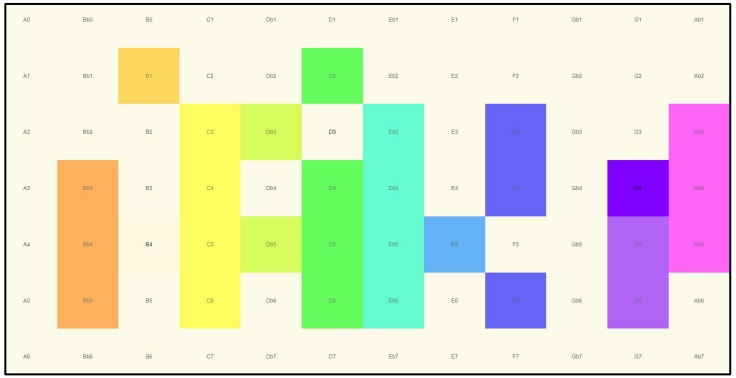
3 components

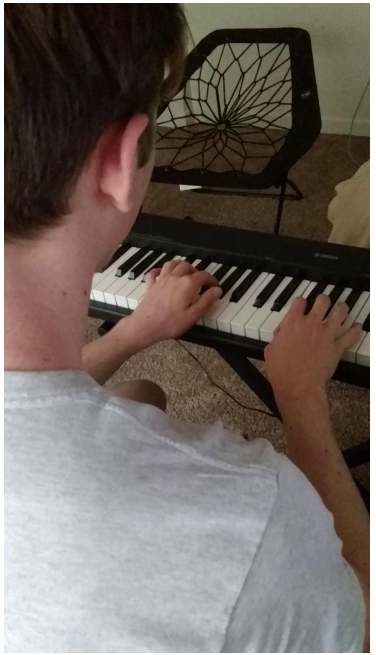
- Grid keyboard that reads MIDI input, plays touched notes
- Bar chart (D3) that logs note frequency
- Media player that plays preset or uploaded MIDI files

\+ 88 additional individual note windows (e.g. C4, A0, G5)

Stored data is note frequency with custom events sent via Sockets.







Airplane flight dashboard

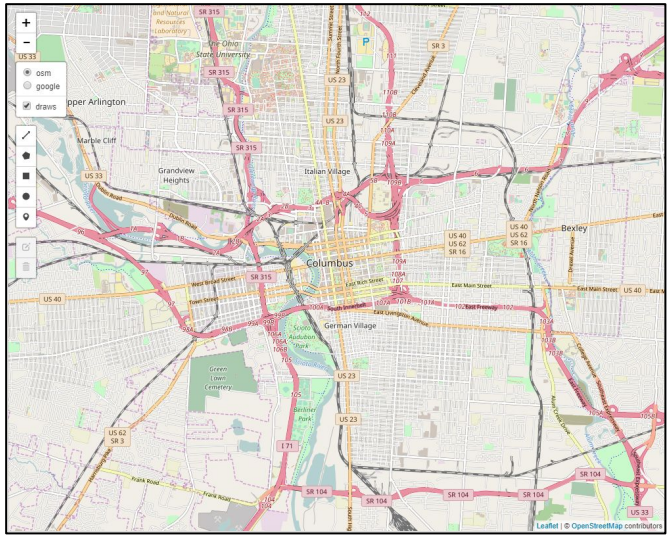
- Several components from the **Square Crossfilter demo**
- Added a **secondary slider dashboard** (2 separate windows)

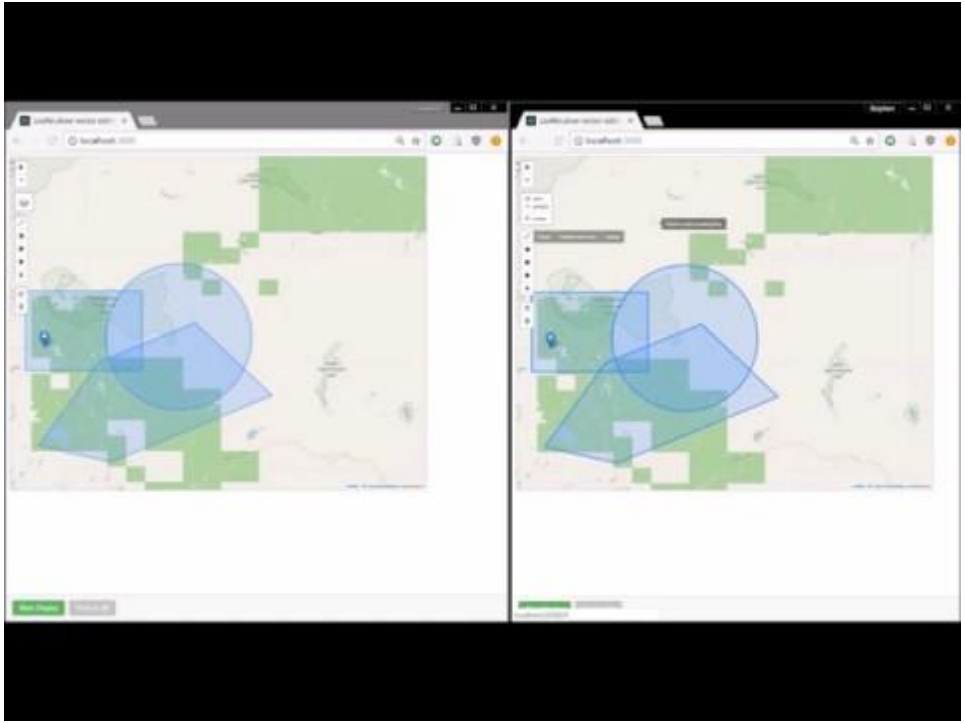
Sliders were manually coded, but this is something that could possibly be generated in the future.

Stored data were the extents of each filter.

Collaborative map

Leaflet Draw demo integrated with Chorus. **Stored data includes map bounds, zoom level, annotations.**

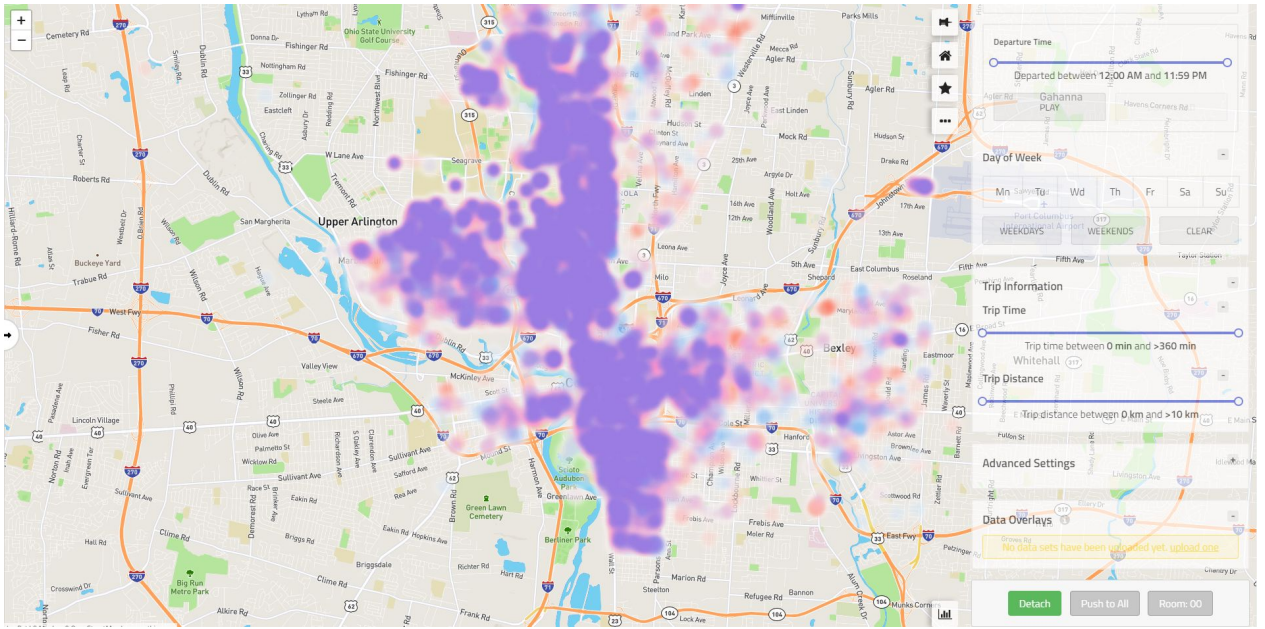




Crossroads Integration

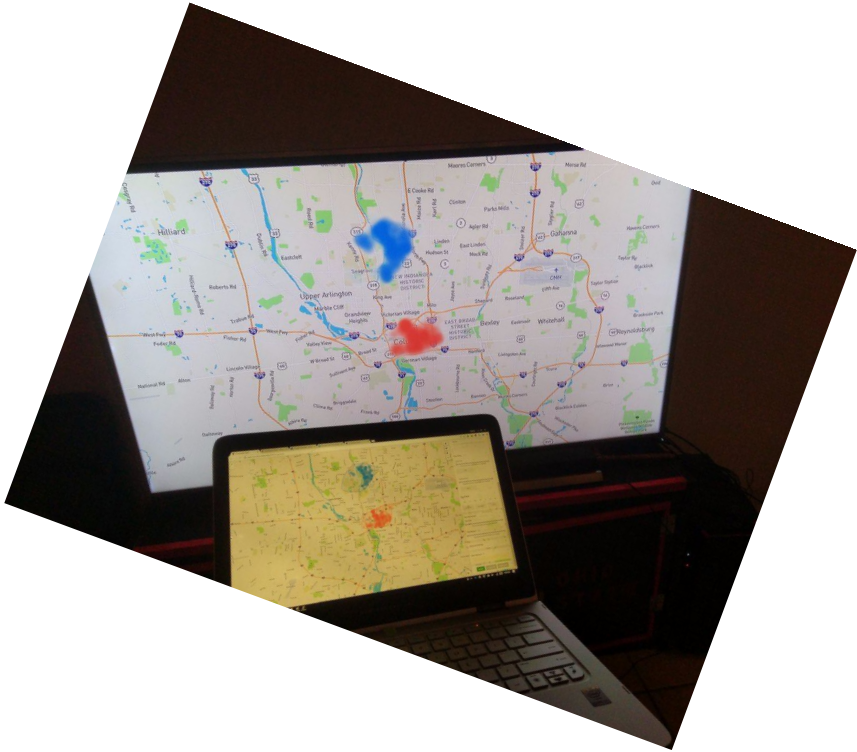
Pretty easy to set up with Dan and Trey's Crossroads project, <30 LOC.

Some issues with un-serializable objects (like Moment.JS dates) that can be easily fixed with JSON stringify / parse



Chromecast

- **Goal**: Use Chromecast + Chorus as a $35 portable viz display, that stays synced to Main Display and has real-time updates from other actors



- Tested with Leaflet demo, works with resolution issues
  - Chromecast has its own internal styling, including a default black background, that it adds via JS
- Debugging is annoying because you need a WEP/WPA(2) phone hotspot
- Seems to have sufficient Javascript compatibility, but more testing to be done

What's Next

- **Documentation**
- **Client Improvements** — Improvements for "clients" (viz developers) to adopt
- **Styling** — Improved styling, especially for Chromecast
- **Security** — Malicious user can manipulate data in shady ways
- **Code Review** — Code is pretty hacky in some areas, can be more object-oriented in some cases
- **Optimization** — Reduce socket events, re-renders, etc.

- Ports are getting opened on **capri.cse.ohio-state.edu** for live demos...just waiting on approval