

# Mapbox GL

Modern, Robust, Open Source  
Maps on Mobile & Web

*Justin Miller • @incanus77*



I did a talk like this at OSB '15, but today we've realized the promise of a lot of this.

Plus, I'd like to talk a little about the project and its foundations.

How many folks are familiar with either Mapbox or Mapbox GL?

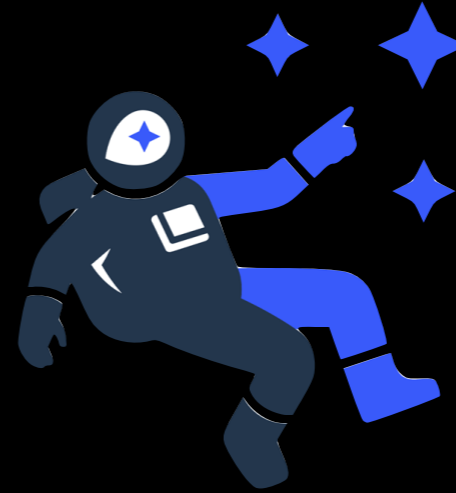
# Introduction

- I do mobile engineering & outreach at Mapbox
- My background is in sysadmin/devops, UNIX, PHP, Mac/iOS, Android, C++
- I want to talk today about project/product divide, a technical overview, & sustaining open source



# Mapbox

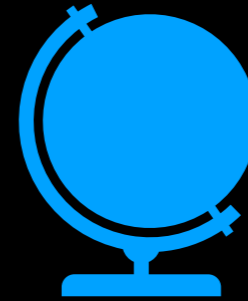
- Building open source tools for custom map design & development
- Cloud hosting of data backend (increasingly realtime)
- About 250 folks worldwide
- Yes, we are for-profit, but also open source



Realtime backend: OSM QA, telemetry & Pulse, satellite

# Maps for Maps' Sake

- Google Maps for Android, iOS, and web
  - Maps as a byproduct of advertising data
- Apple Maps for iOS and macOS
  - Maps as a byproduct of hardware sales
- Mapbox for Android, iOS, web, macOS, QT, C++, Unity...
  - Maps for maps' sake as a flexible toolkit/platform



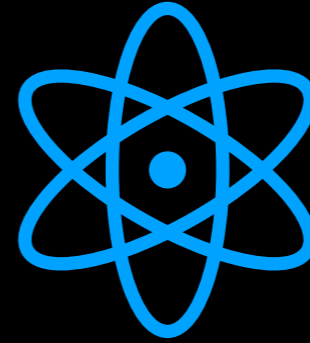
# Mapbox GL

- Renderer technology/project name
- `mapbox-gl-native` (C++/Objective-C/Java)
- `mapbox-gl-js` (uhhh... JS)
- Forms basis of mobile native & JavaScript SDKs



# Tech Core

- OpenGL (OpenGL ES on mobile, WebGL in HTML)
  - Hardware-accelerated, built on standards, fast
- Tries not to compromise
  - Built at the lowest possible levels
- It's a *platform* which provides & relies on more standards



Compromises: hybrid/cross-platform frameworks, need to build the root anyway

Styling: primitives, icons/sprites, metadata, data sources, layering

# Mapbox GL Native

The screenshot shows the GitHub repository page for `mapbox/mapbox-gl-native`. The repository is described as "Interactive, thoroughly customizable maps in native Android, iOS, macOS, Node.js, and Qt applications, powered by vector tiles and OpenGL". It has 1,940 stars, 600 forks, and 239 watchers. The repository contains 10,766 commits, 166 branches, 371 releases, and 117 contributors. The current branch is `master`. The commit history shows the following entries:

Commit	Message	Time
<code>jfirebaugh [core]</code>	Move <code>setStyle.JSON/URL</code> to <code>Style</code> ; add <code>Map::setStyle</code>	Latest commit 40e4755 2 days ago
<code>[ios]</code>	Configured <code>stringsdict</code> from Transifex	2 months ago
<code>[core]</code>	Move <code>setStyle.JSON/URL</code> to <code>Style</code> ; add <code>Map::setStyle</code>	an hour ago
<code>[core]</code>	Move <code>setStyle.JSON/URL</code> to <code>Style</code> ; add <code>Map::setStyle</code>	an hour ago
<code>[build]</code>	Give <code>ios/builds</code> <code>s3</code> permission to Bitrise user	3 months ago
<code>[all]</code>	Promote <code>Style</code> to public API	an hour ago
<code>[qt]</code>	Bring back <code>icon.png</code>	2 months ago

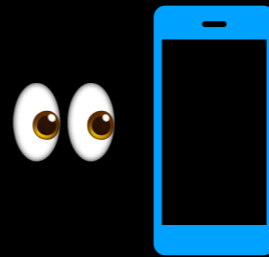
# Mapbox GL JS

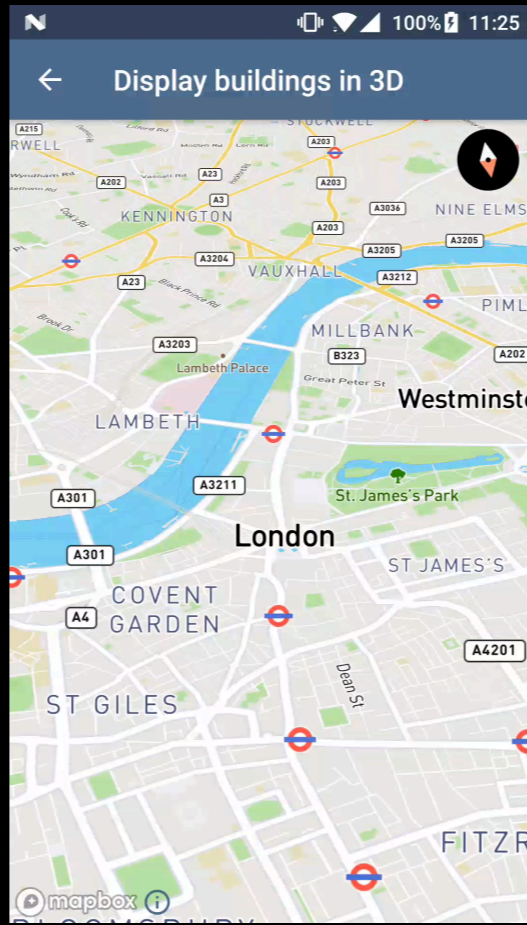
The screenshot shows the GitHub repository page for Mapbox GL JS. The repository is owned by mapbox and has 1,869 stars, 246 forks, and 442 watchers. It has 521 issues, 28 pull requests, and 5 projects. The repository description is "Interactive, thoroughly customizable maps in the browser, powered by vector tiles and WebGL" with the URL <https://www.mapbox.com/maps>. The repository statistics show 6,887 commits, 60 branches, 90 releases, and 147 contributors. The current branch is master. The commit history shows the latest commit by anandthakker 2 hours ago, with a list of recent commits and their descriptions.

Commit	Description	Time Ago
anandthakker	Enhance render test results page (#4868)	2 hours ago
.github	Move github templates into .github/ (#4806)	14 days ago
bench	Add links to run individual benchmarks	29 days ago
ci-scripts	make sure css is linted on CI	2 months ago
cloudformation	deploy to CN regions (#2915)	11 months ago
debug	Add support to show and track user location in the GeolocateControl (#...	8 days ago
dist	Add support to show and track user location in the GeolocateControl (#...	8 days ago
docs	Update docs and test to newer version of mapbox-gl-rtl-text plugin.	6 days ago

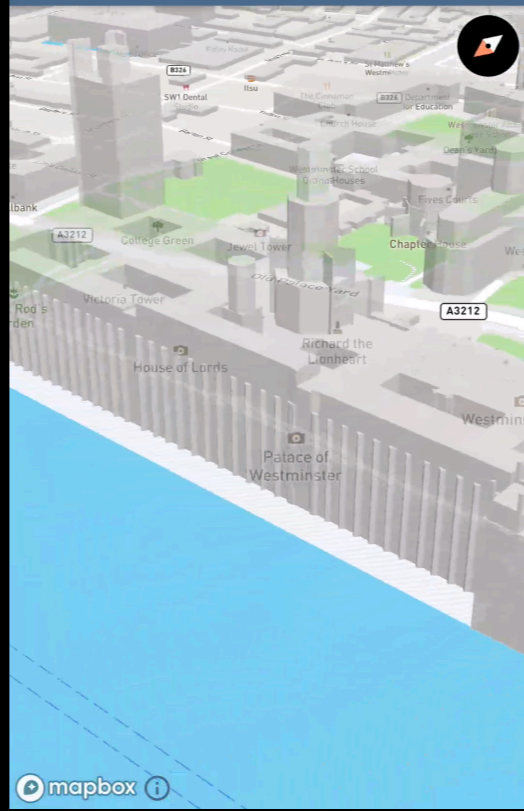


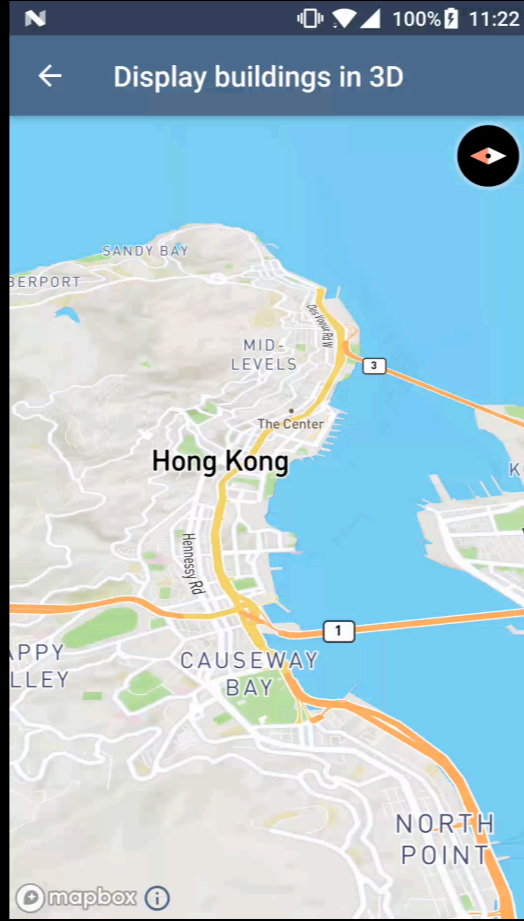
Enough talk, let's see  
a couple demos!

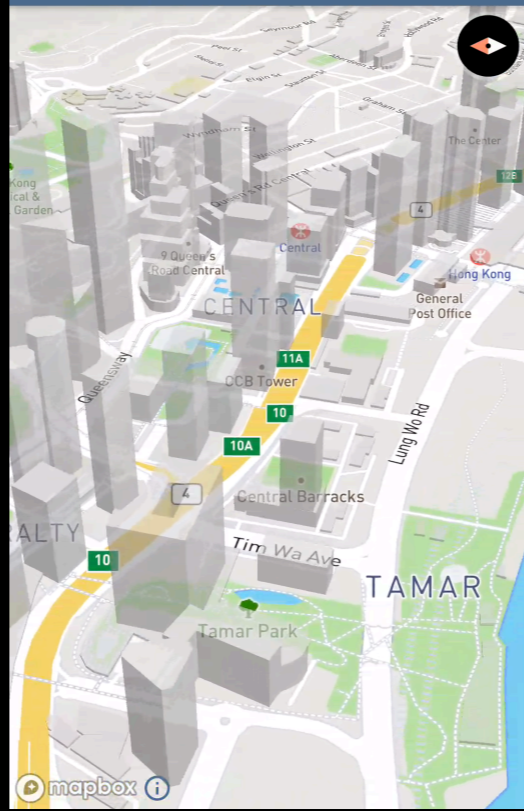




# ← Display buildings in 3D











### Display 3D building height based on data

Use extrusions to display 3D building height based on imported data

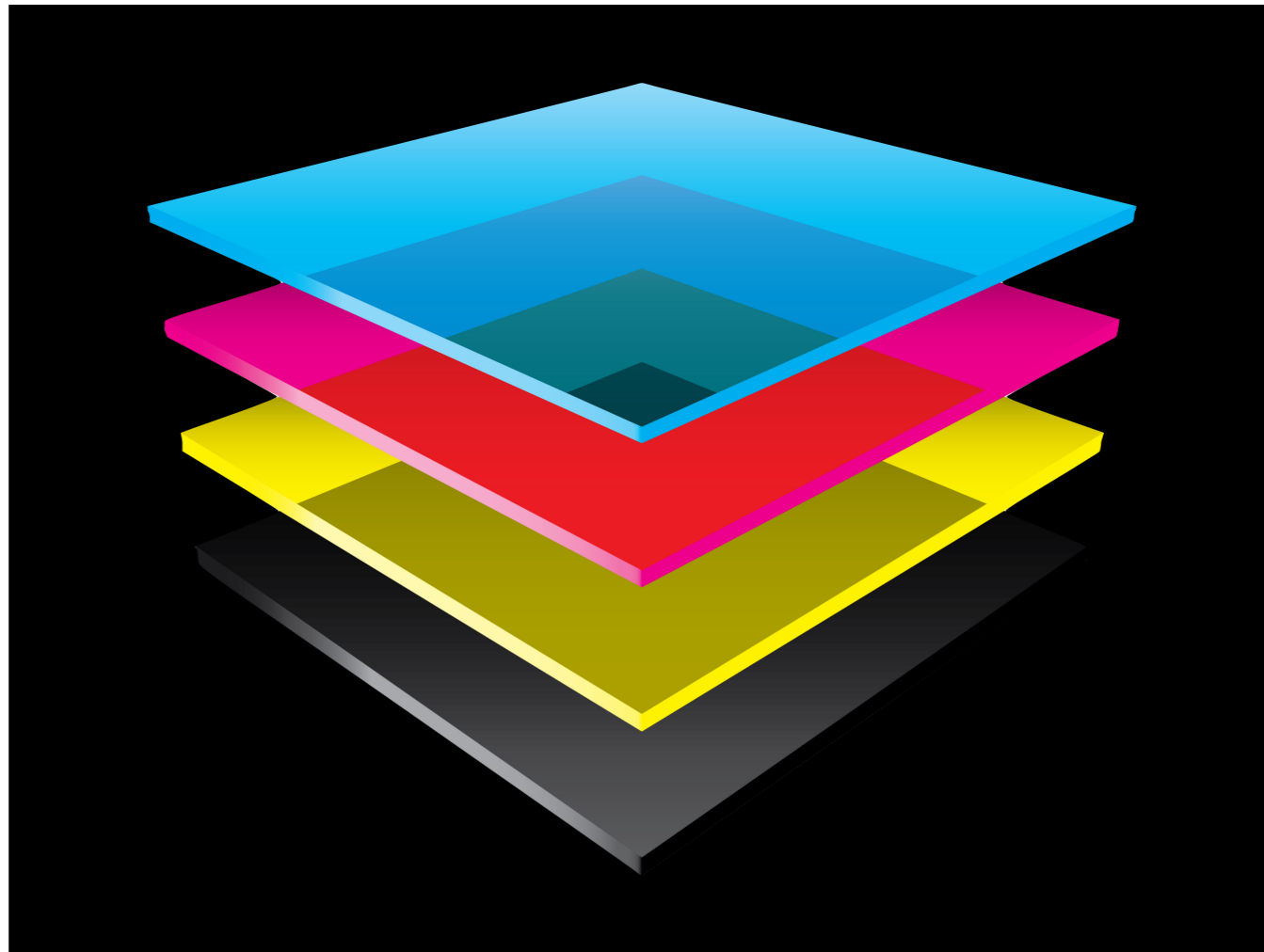


### Display elevation and location data

Use extrusions to display elevation data along a route



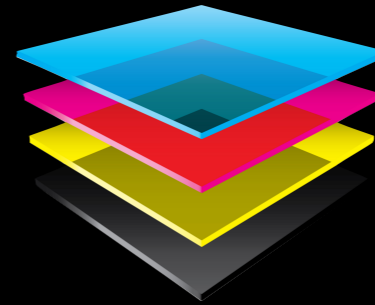




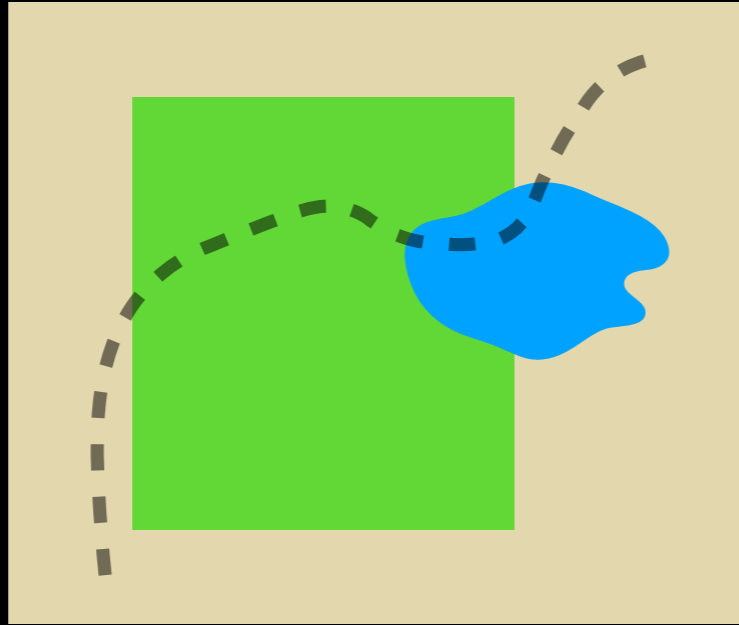
Maps are about layers. We like to think of maps as layers for editing, not stuff you put your stuff on top of.

# Layers Have Stuff

- There are types of layers (i.e. primitives)
- Layers have properties
- Properties have different types of values
- Layers are ordered



# line Layer



- `line-color`
- `line-width`
- `line-dasharray`
- `line-opacity`
- `line-cap`
- `line-translate`
- ...

apologize/survey for color-blindness

# Style Spec

- Defines all of this for many layer types (`line`, `fill`, `symbol`, `raster`, `fill-extrusion...`)
- This is an open spec (now part of GL JS repo)
- Has bindings for various languages
  - JavaScript, C++
- Spec available as Markdown



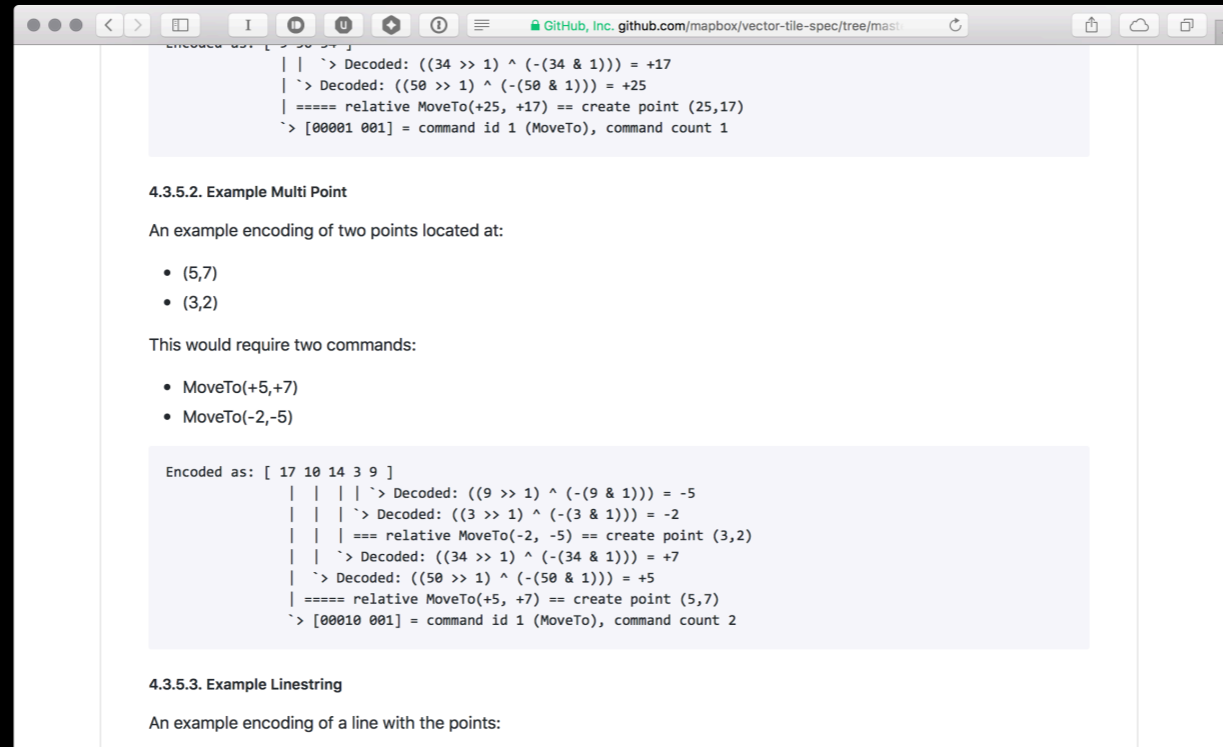
# Mapbox GL Style Spec

```
479     "android": "2.0.1",
480     "ios": "2.0.0",
481     "macos": "0.1.0"
482   }
483 }
484 },
485 },
486 "layout_circle": {
487   "visibility": {
488     "type": "enum",
489     "values": {
490       "visible": {
491         "doc": "The layer is shown."
492       },
493       "none": {
494         "doc": "The layer is not shown."
495       }
496     },
497     "default": "visible",
498     "doc": "Whether this layer is displayed.",
499     "sdk-support": {
500       "basic functionality": {
501         "js": "0.10.0",
502         "android": "2.0.1",
503         "ios": "2.0.0",
504         "macos": "0.1.0"
505       }
506     }
507   },
508 },
509 "layout_fill-extrusion": {
510   "visibility": {
511     "type": "enum",
```

# What About Data?

- Vector tile format (also open as `vector-tile-spec`)
  - Maps are tiled (think: checkerboards for each scale)
- Concept of "data layers"
  - Points of interest (names & types as symbols)
  - Roads (names as symbols & geometries as lines)
  - Parks (geometries as fills)

# Vector Tile Spec



```
Encoded as: [ 9 30 27 ]
| | `> Decoded: ((34 >> 1) ^ (-(34 & 1))) = +17
| `> Decoded: ((50 >> 1) ^ (-(50 & 1))) = +25
| ===== relative MoveTo(+25, +17) == create point (25,17)
| `> [0001 001] = command id 1 (MoveTo), command count 1
```

**4.3.5.2. Example Multi Point**

An example encoding of two points located at:

- (5,7)
- (3,2)

This would require two commands:

- MoveTo(+5,+7)
- MoveTo(-2,-5)

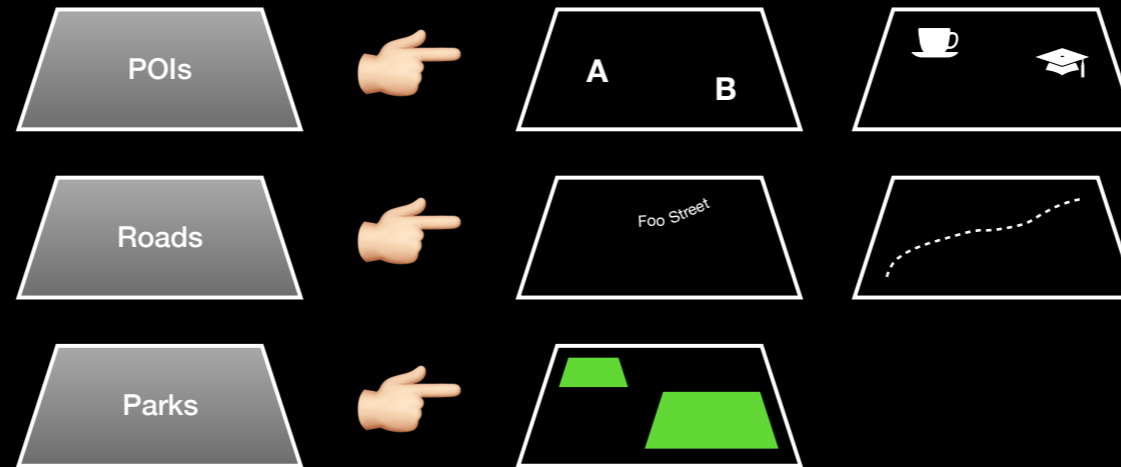
```
Encoded as: [ 17 10 14 3 9 ]
| | | | `> Decoded: ((9 >> 1) ^ (-(9 & 1))) = -5
| | | | `> Decoded: ((3 >> 1) ^ (-(3 & 1))) = -2
| | | == relative MoveTo(-2, -5) == create point (3,2)
| | `> Decoded: ((34 >> 1) ^ (-(34 & 1))) = +7
| | `> Decoded: ((50 >> 1) ^ (-(50 & 1))) = +5
| ===== relative MoveTo(+5, +7) == create point (5,7)
| `> [0010 001] = command id 1 (MoveTo), command count 2
```

**4.3.5.3. Example Linestring**

An example encoding of a line with the points:



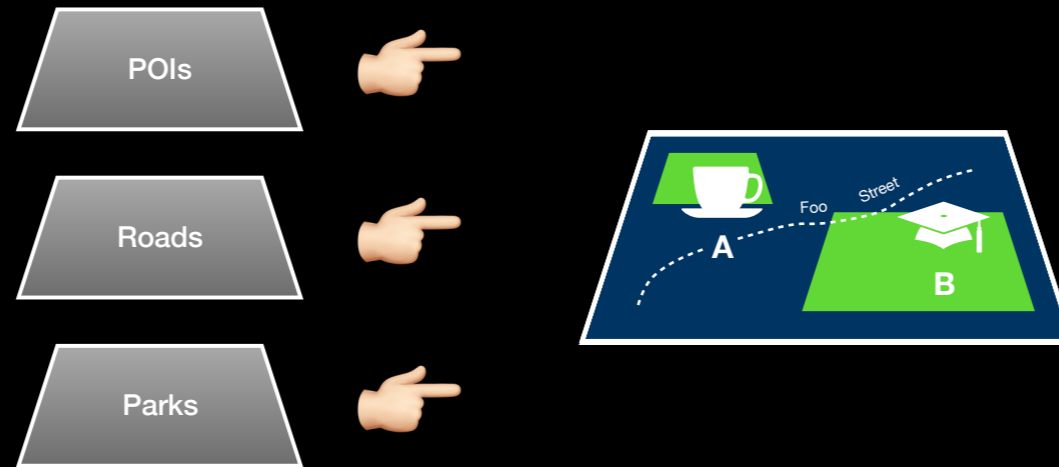
# Data Layers 🙋 Style Layers



The cool thing here is the data layer order doesn't matter.

In fact, you can interleave different data sources in the way you order the style layers.

# Data Layers 🙌 Style Layers

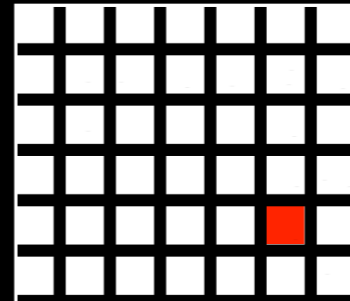


The cool thing here is the data layer order doesn't matter.

In fact, you can interleave different data sources in the way you order the style layers.

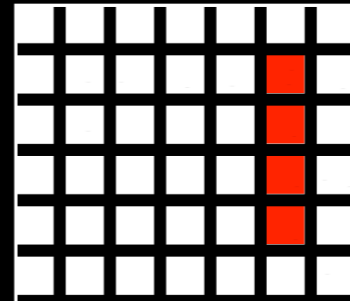
# Vector Tiles Are Just Grids

- Not mathematical vectors (e.g. beziers)
- Rather they are a 4096 resolution grid
- Vector geometries are encoded as delta drawing steps
  - move to 5, 4
  - draw up 3
  - draw left 4
  - draw down 2



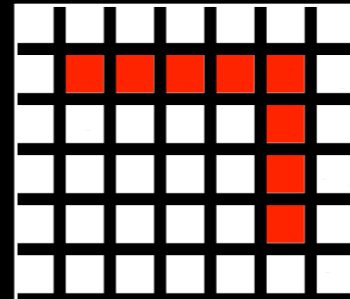
# Vector Tiles Are Just Grids

- Not mathematical vectors (e.g. beziers)
- Rather they are a 4096 resolution grid
- Vector geometries are encoded as delta drawing steps
  - move to 5, 4
  - draw up 3
  - draw left 4
  - draw down 2



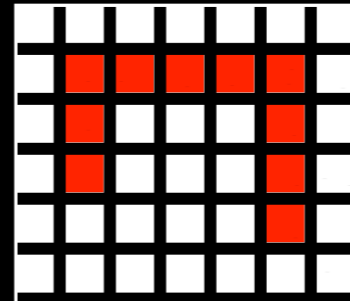
# Vector Tiles Are Just Grids

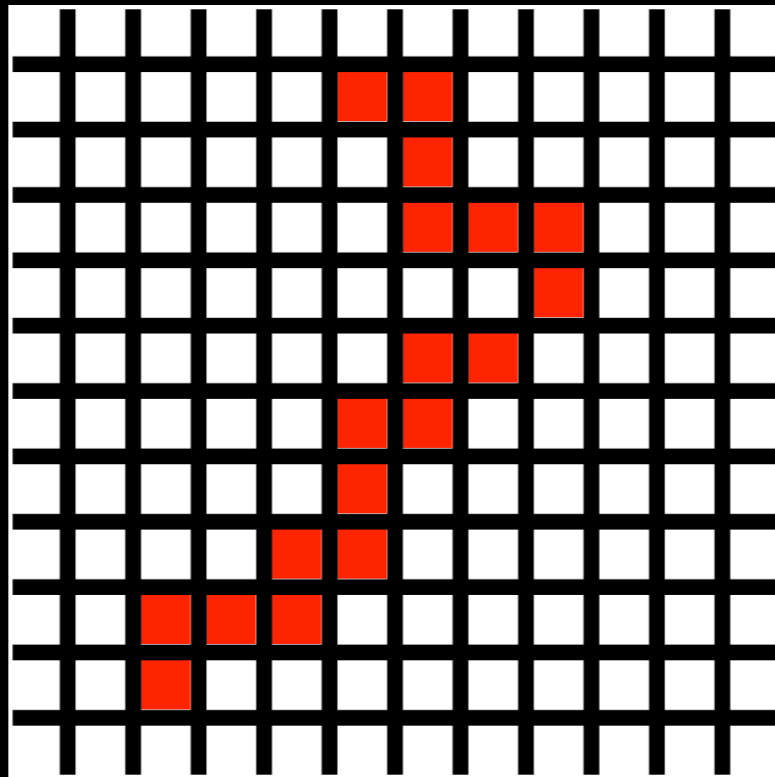
- Not mathematical vectors (e.g. beziers)
- Rather they are a 4096 resolution grid
- Vector geometries are encoded as delta drawing steps
  - move to 5, 4
  - draw up 3
  - draw left 4
  - draw down 2



# Vector Tiles Are Just Grids

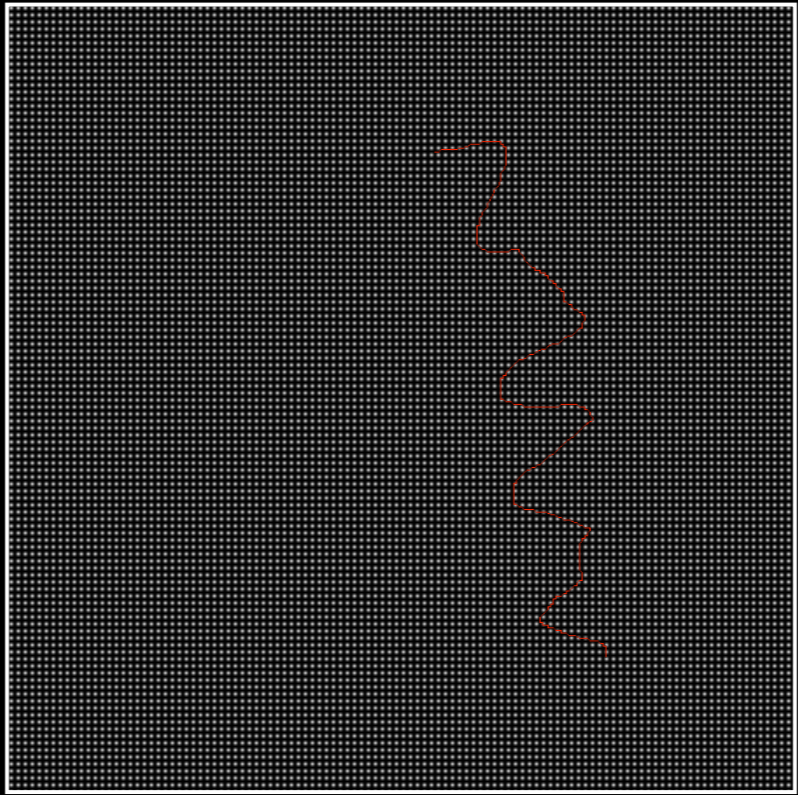
- Not mathematical vectors (e.g. beziers)
- Rather they are a 4096 resolution grid
- Vector geometries are encoded as delta drawing steps
  - move to 5, 4
  - draw up 3
  - draw left 4
  - draw down 2





12

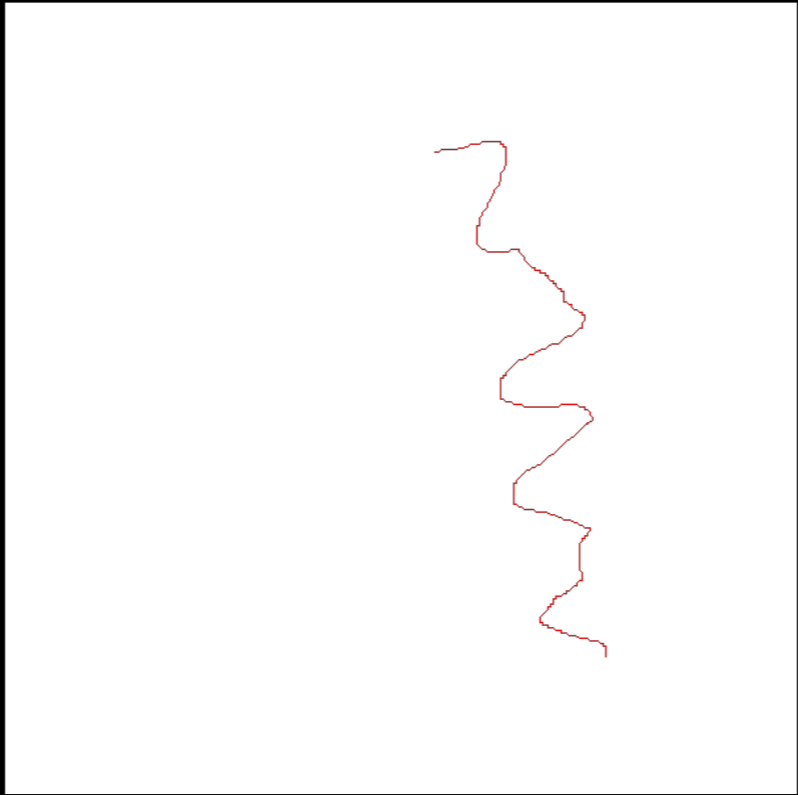
12



4096

4096





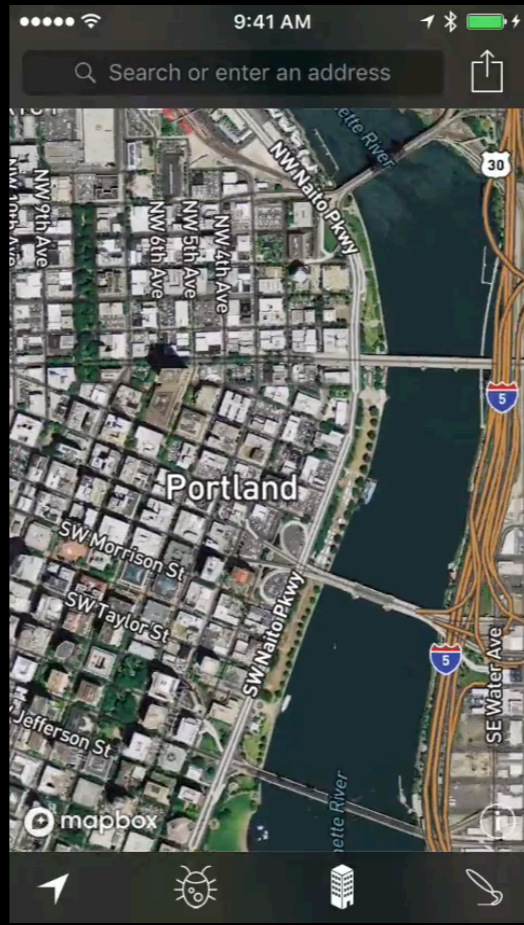
4096

4096

# True Rasters are Still Supported!

- Satellite imagery is the big use case
- Rasters are able to be sandwiched with vector layers
- Example: “Satellite Streets”



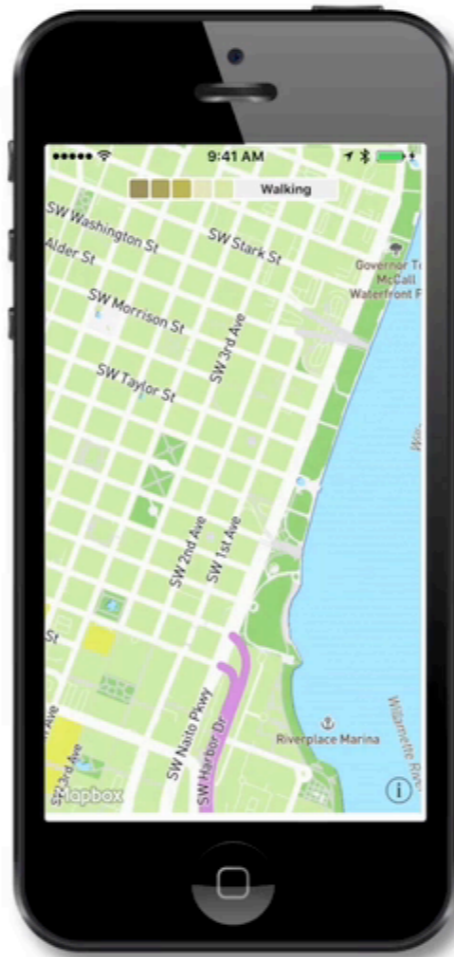


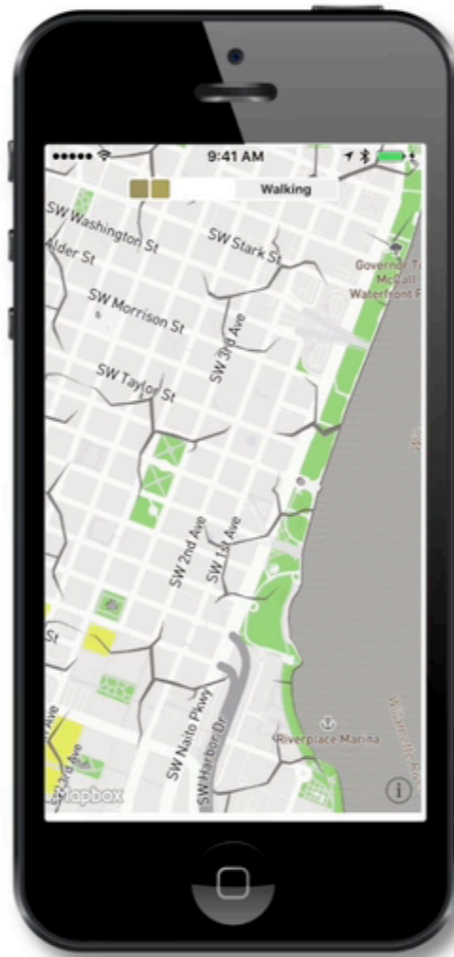


# Runtime Styling!

- All of these properties can be changed at runtime!
  - Example: change parks from green to brown
- In JavaScript, the style is literally JSON, so just mutate it
- In native, there are strongly-typed APIs

```
map.style.layer(withIdentifier: "parks").fillColor =  
    MGLStyleValue(rawValue: .brown)
```







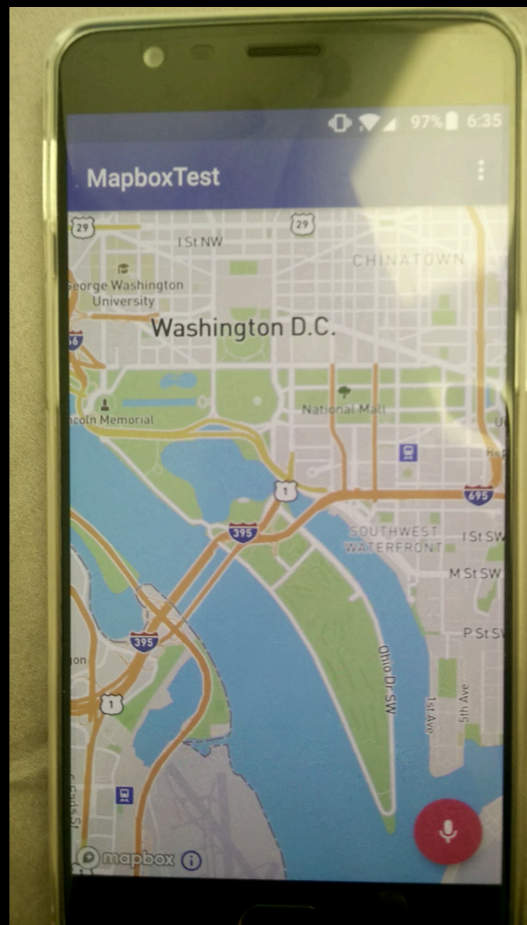


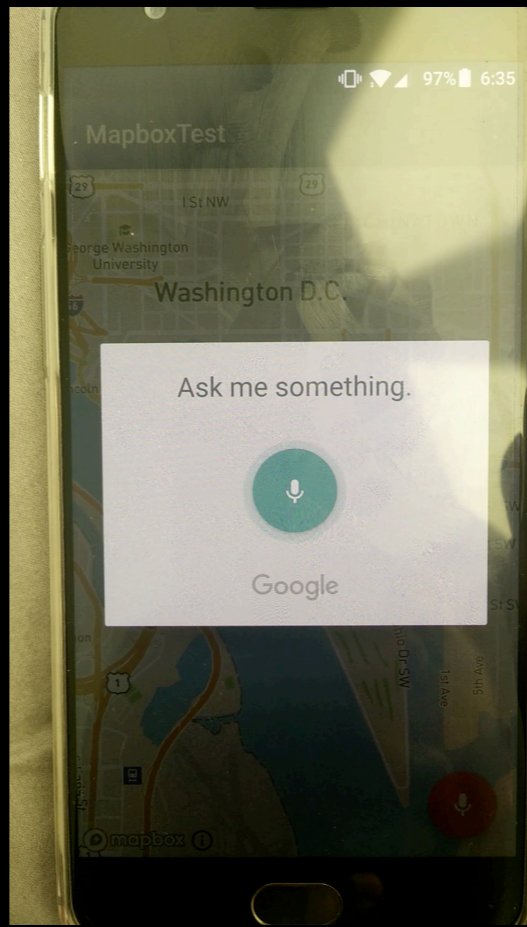


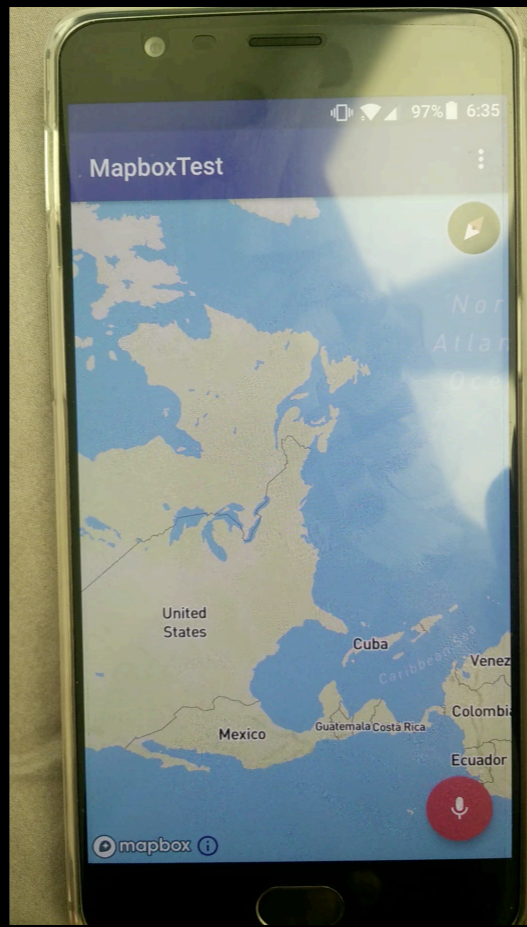


# Movement & Animation!

- Transitions
  - For every e.g. `fill-color`, there is a `fill-color-transition`
  - Has a `delay` and/or a `duration`
  - Example: “transition to red 1.0s from now over 2.0s”
- Animated camera/viewport
  - Pretend like you are in a helicopter!



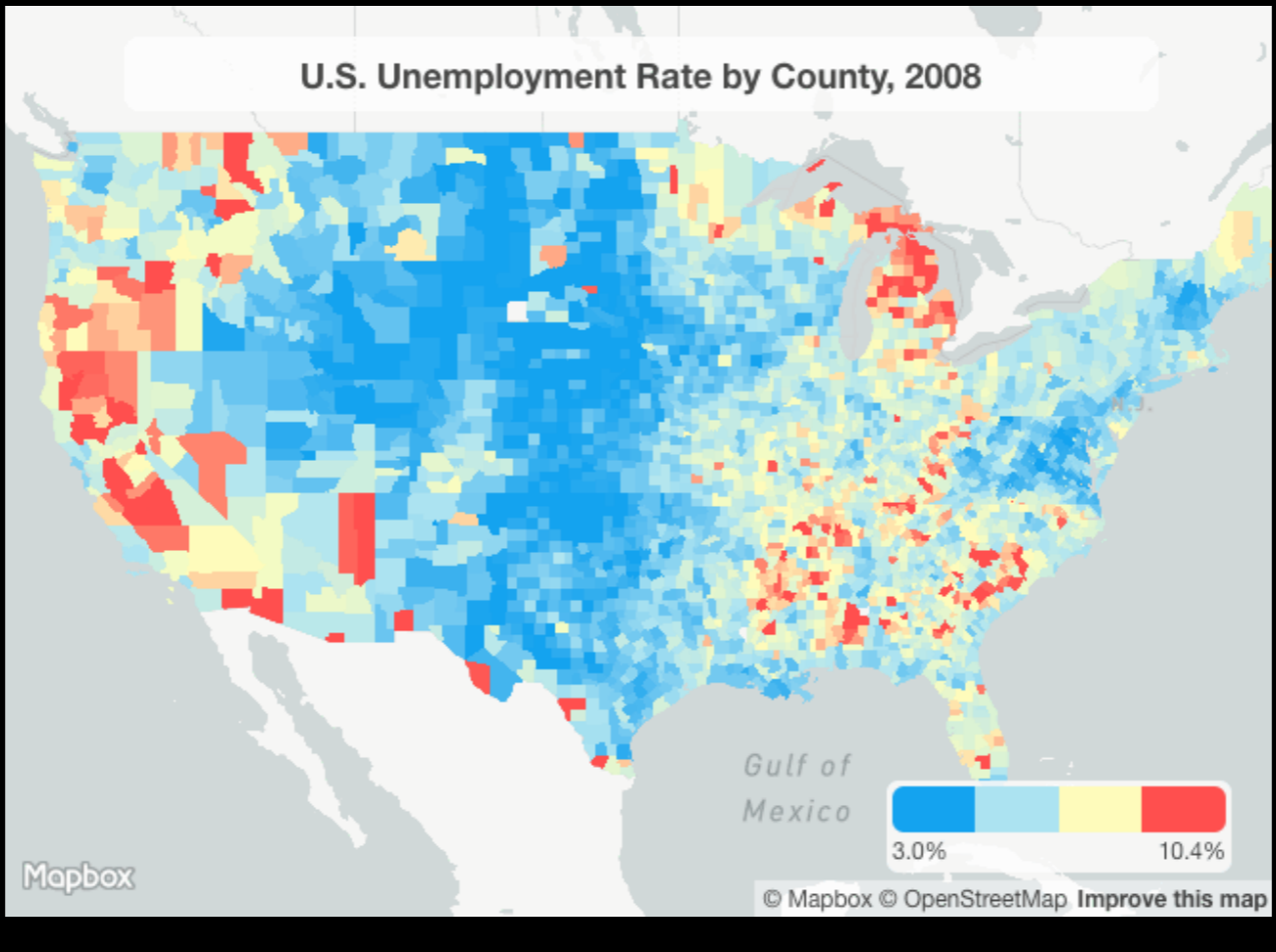




# One Further: Data-Driven or Property Styling

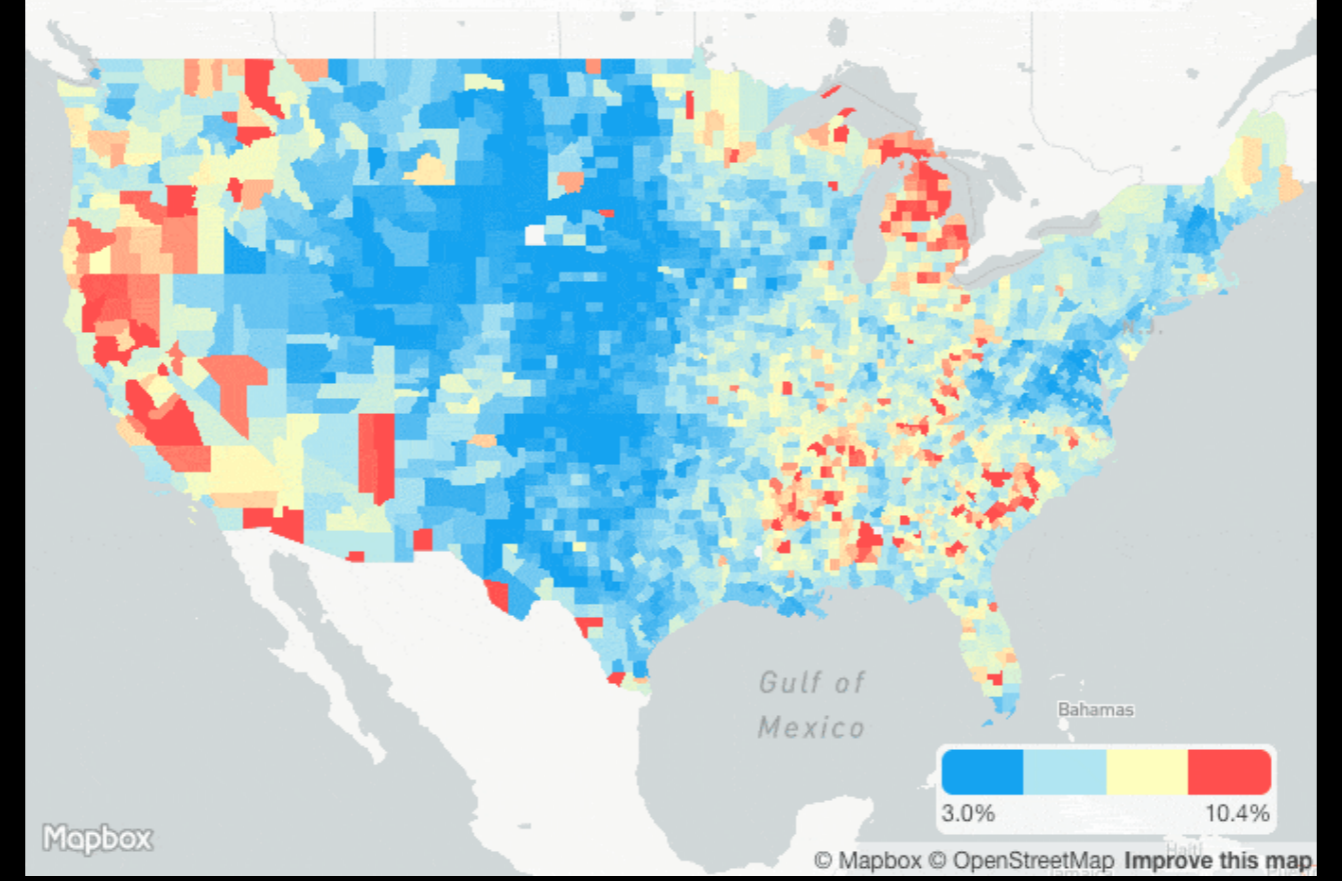
- Example: height of individual building tied to `{height}` field (exponential/linear)
- Example: icon of POI tied to POI's `{type}` field (identity)
- Example: color of county tied to one of four values based on `{unemployment}` field range (categorical)

U.S. Unemployment Rate by County, 2008

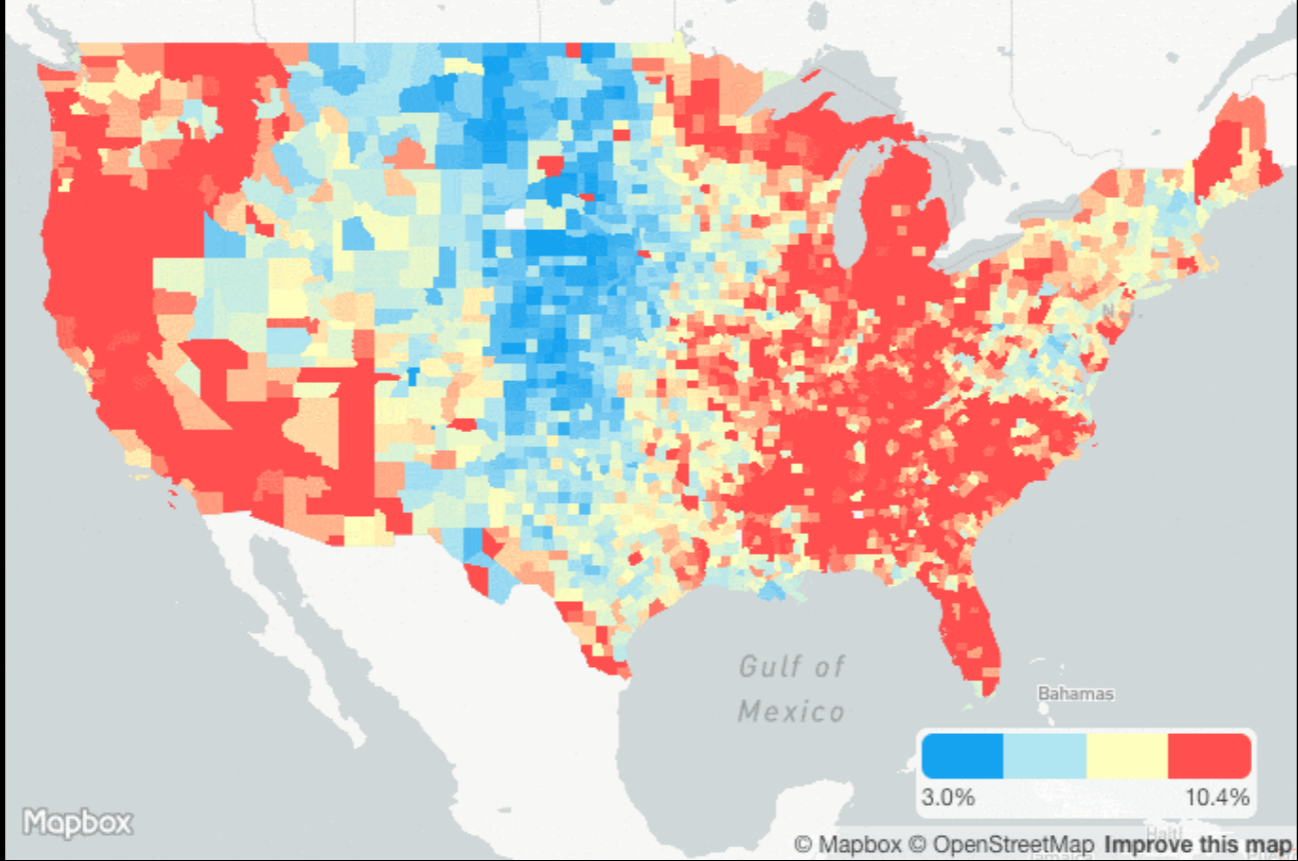




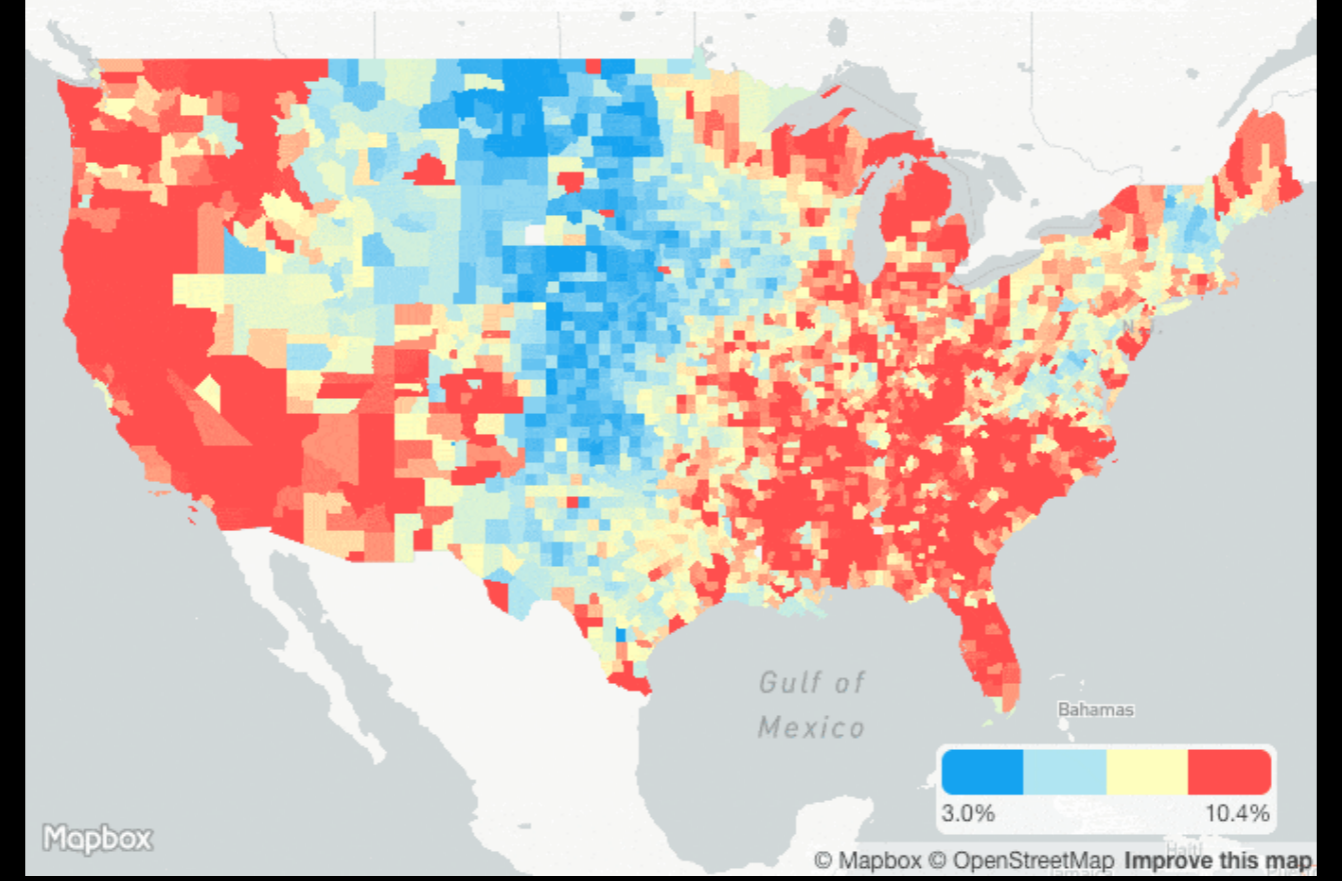
# U.S. Unemployment Rate by County, 2008



U.S. Unemployment Rate by County, 2009



U.S. Unemployment Rate by County, 2011



# Challenges of Open Source

- Edge case features
  - Discussion, influence, rejection
  - Plugins
- Customer strategy
  - One-way GitHub linking from private repos
- We don't throw code over the wall
  - Mistakes are public
  - Walkbacks are public
  - Indecision is public
- But it's real—software isn't perfect

# Public Repositories

- `https://github.com/mapbox/...`
  - `mapbox-gl-native`
  - `mapbox-gl-js`
  - `mapbox-gl-style-spec` (in GL JS)
  - `vector-tile-spec`
- And hundreds more! (over 650 currently)



# Recap

- Maps as a platform
- Low-level tech core
- Layers, both style and data
- Open specs
- Runtime styling
- Movement & animation
- Data-driven styling
- Open source real talk

# Thank You!

- [@incanus77](#)
- [justin@mapbox.com](mailto:justin@mapbox.com)
- [@mapbox](#)
- [mapbox.com/blog](https://mapbox.com/blog)

