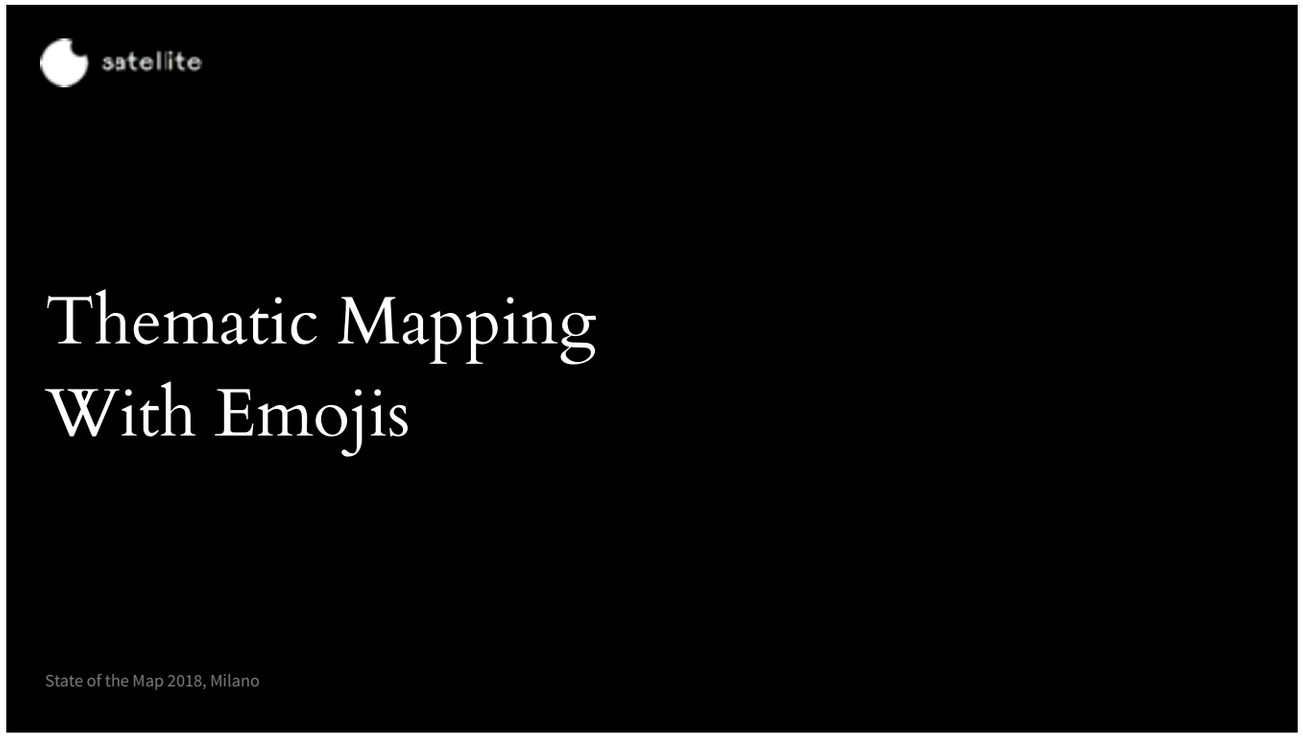
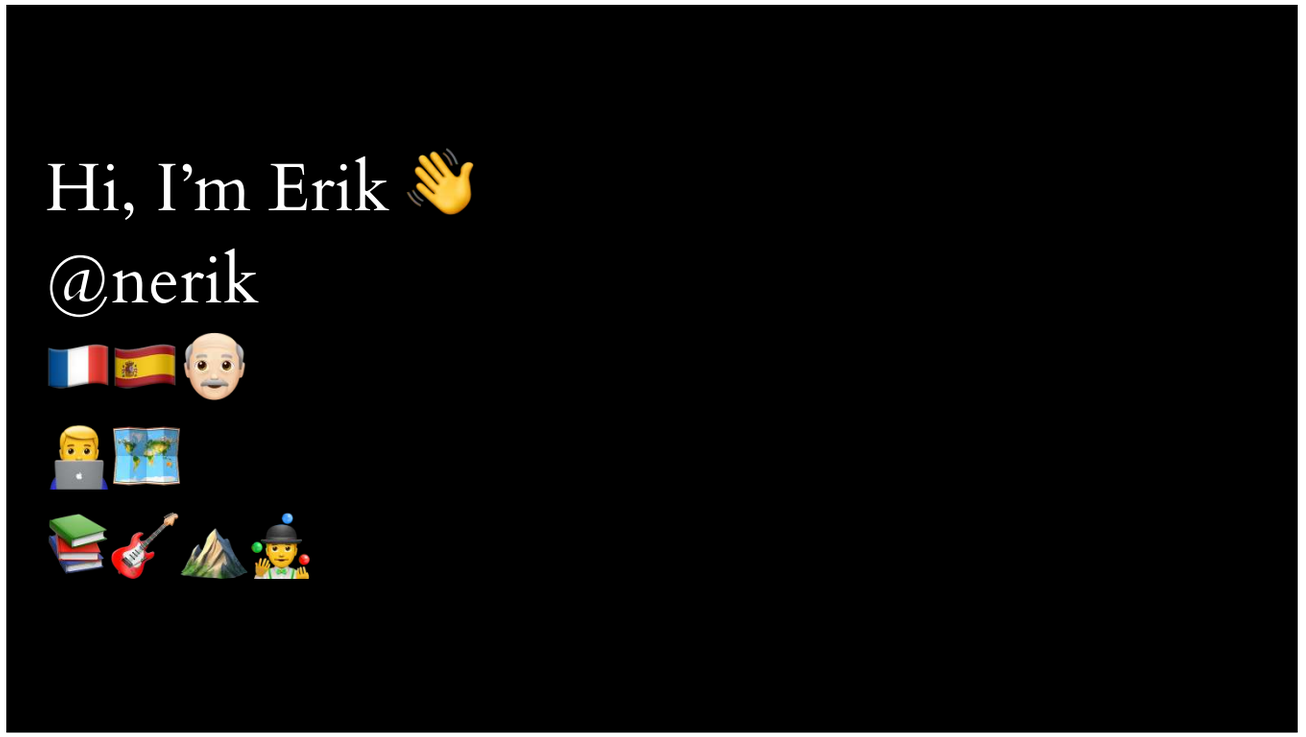


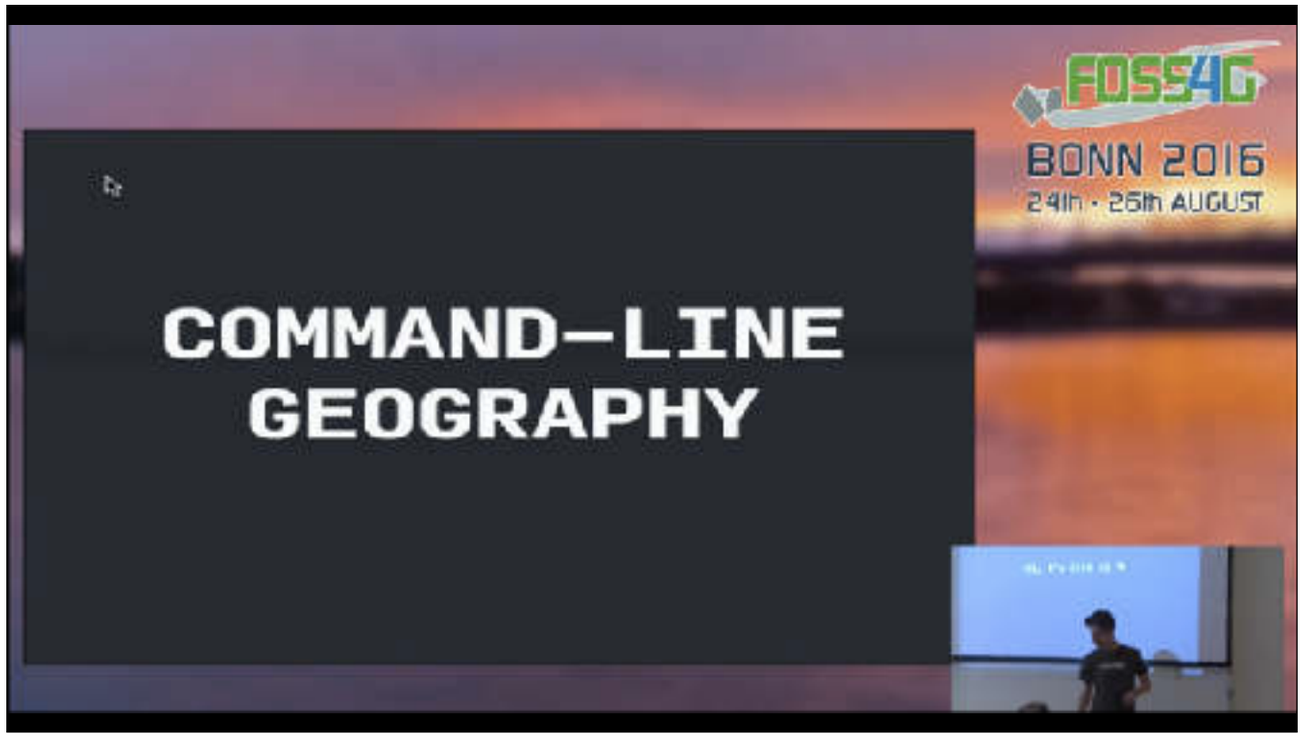
Hi, I’m going to talk about making maps with Emojis.



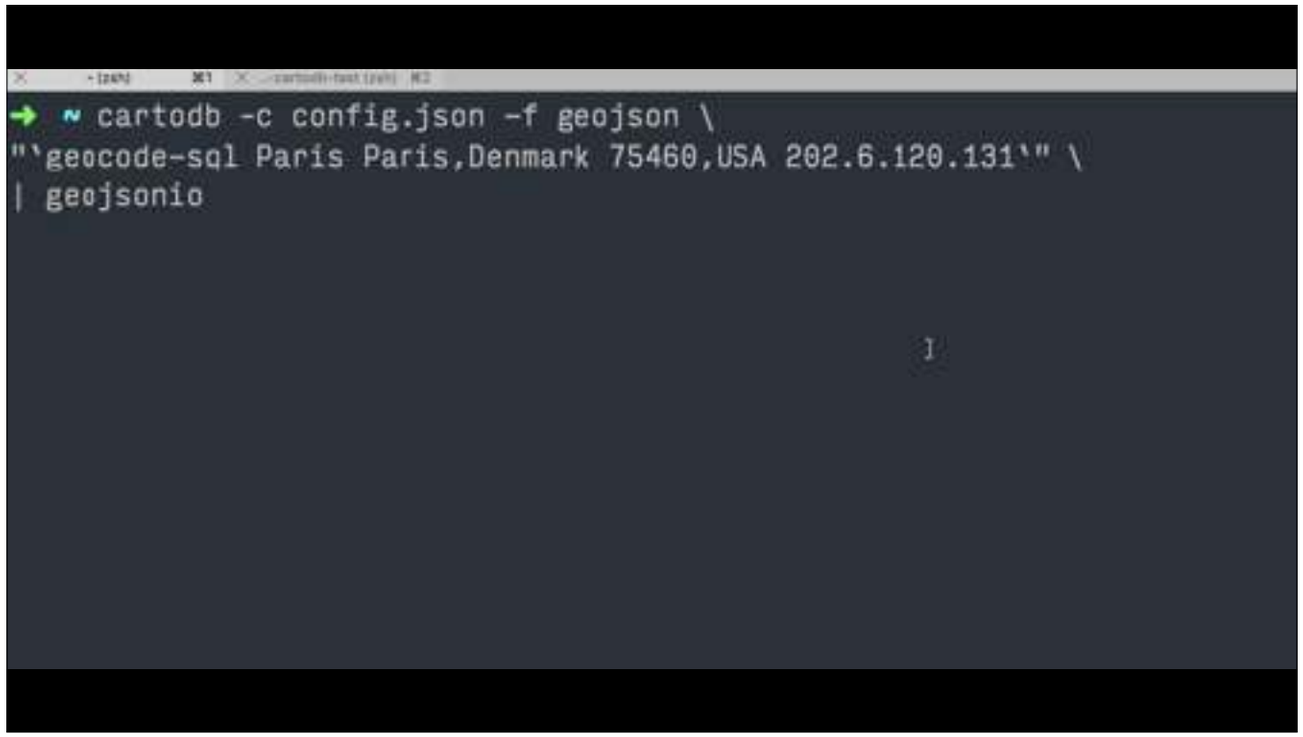
I’m French, living in Madrid, Spain, 34 years old
I’m an engineer, mostly working on interactive maps
I like reading, playing the guitar, going running in the mountains. Juggling is not a hobby at all but I like that there’s an emoji for that (look at the little mustache).

I am the co-founder of Satellite Studio, a small team making interactive maps and interactive infographics.

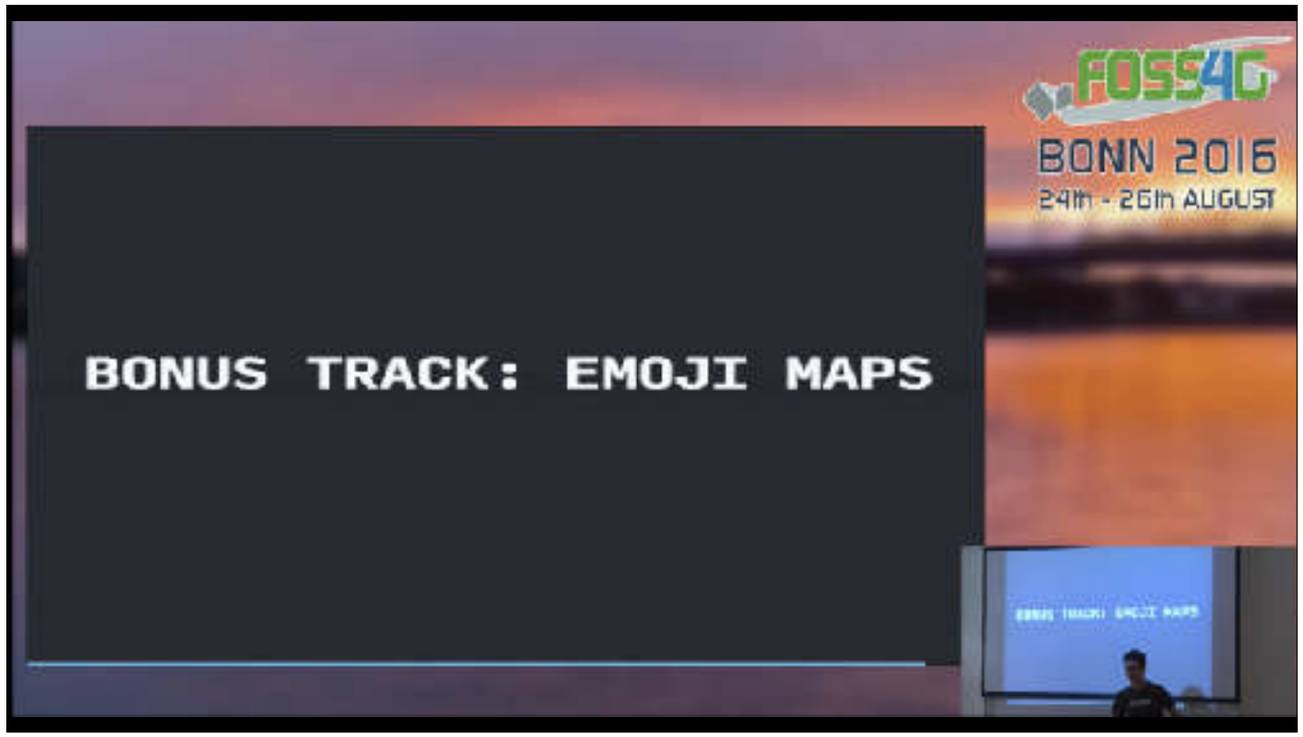
In a past life, I was a “tech evangelist” at Carto - no need to present them here, I guess. I had a great time there, and part of my job was to go to conferences and demo the cool things you can do with the platform.



So in 2016 I presented a talk at FOSS4G in Germany named “Command line geography”.



The idea of this presentation was to show how to take advantage from the Carto APIs to do all sort of geo things in the terminal line
https://ftp.gwdg.de/pub/misc/openstreetmap/FOSS4G-2016/foss4g-2016-1133-command_line_geography-hd.mp4
https://nerik.github.io/cli-geography/



Which got people moderately interested, but then I had a bunch of slides at the end named “Emoji Maps”

```
➜ ~ cartodb -f geojson \
> 'SELECT w.the_geom FROM world_borders wgdp_growth g WHERE w.na
me = g.country.name AND g.gdp_growth_2014 > 2' \
> | gj2ascii --width 200 world.geojson - -c :snowflake -c :smile
:
```

And so you could take a SQL query, get a GeoJSON file and pipe that to a tool named gj2ascii

gj2ascii (GeoJSON to ASCII)

- Renders spatial vectors as ASCII and/or emoji on the command line.
- Written in Python by Kevin Wurster

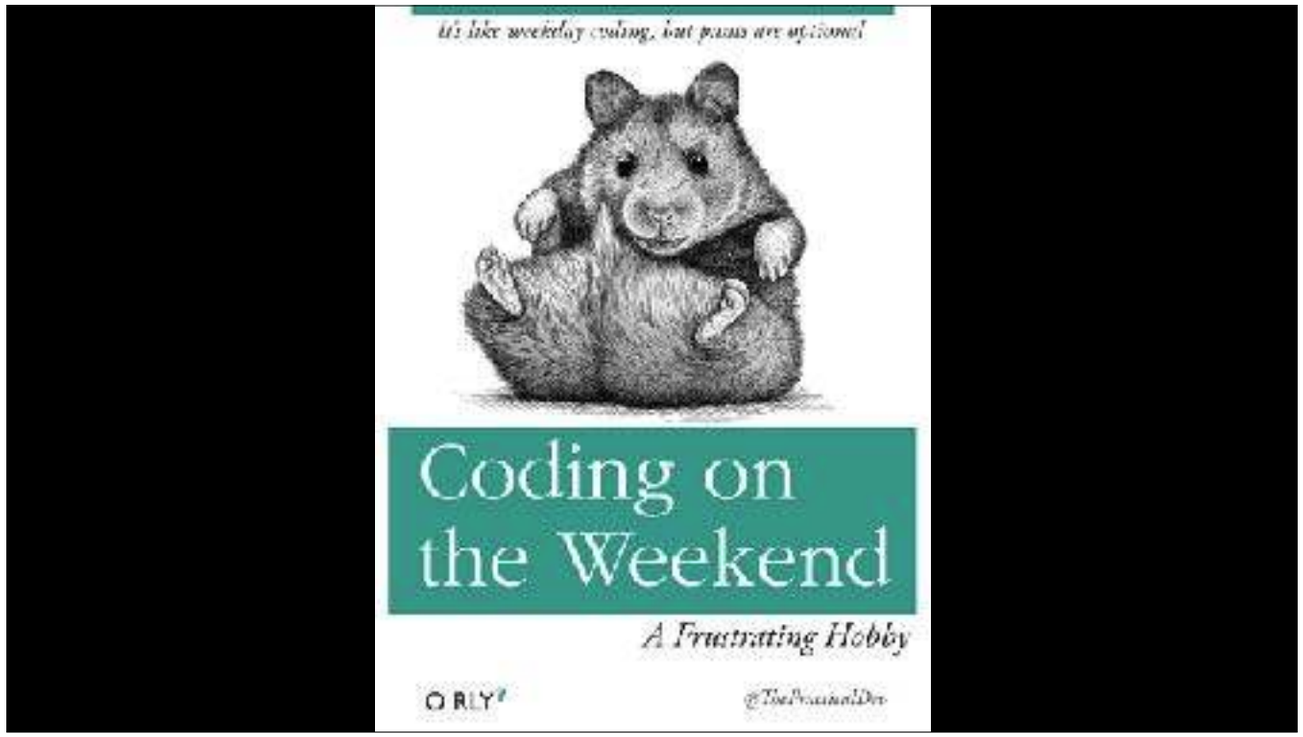
I used a tool I used on the terminal is called gj2ascii and has been the starting point and the main inspiration for that talk.

A world map made of emojis in the terminal (countries with GD growth > 2% get a little smiling face).

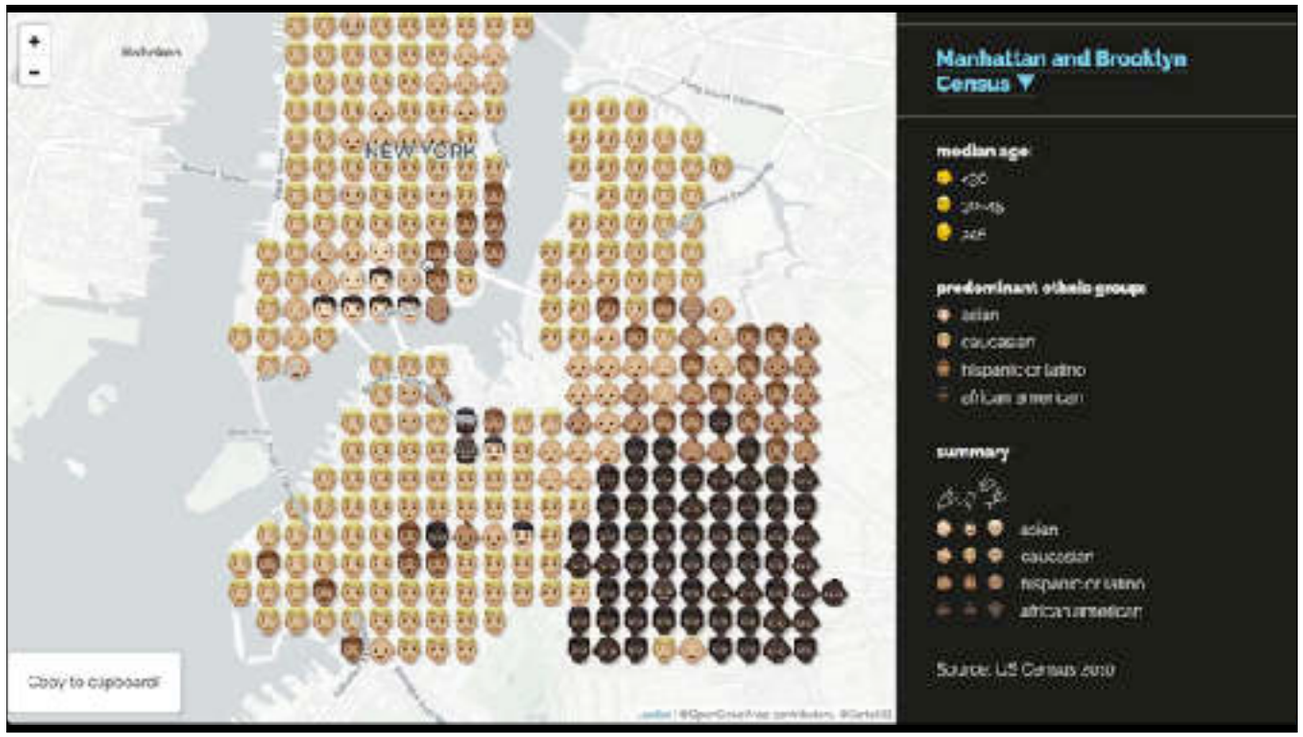
So I got interesting reactions from the FOSS4G folks at the time.

So at this point, I wondered if this might deserve going beyond a quick hack.

And so, it was one of those rainy week-ends in Madrid (we are actually very happy when it happens every 8 months)



so I sat and started coding and working on this emoji maps idea.



And so this was the product of this week-end:

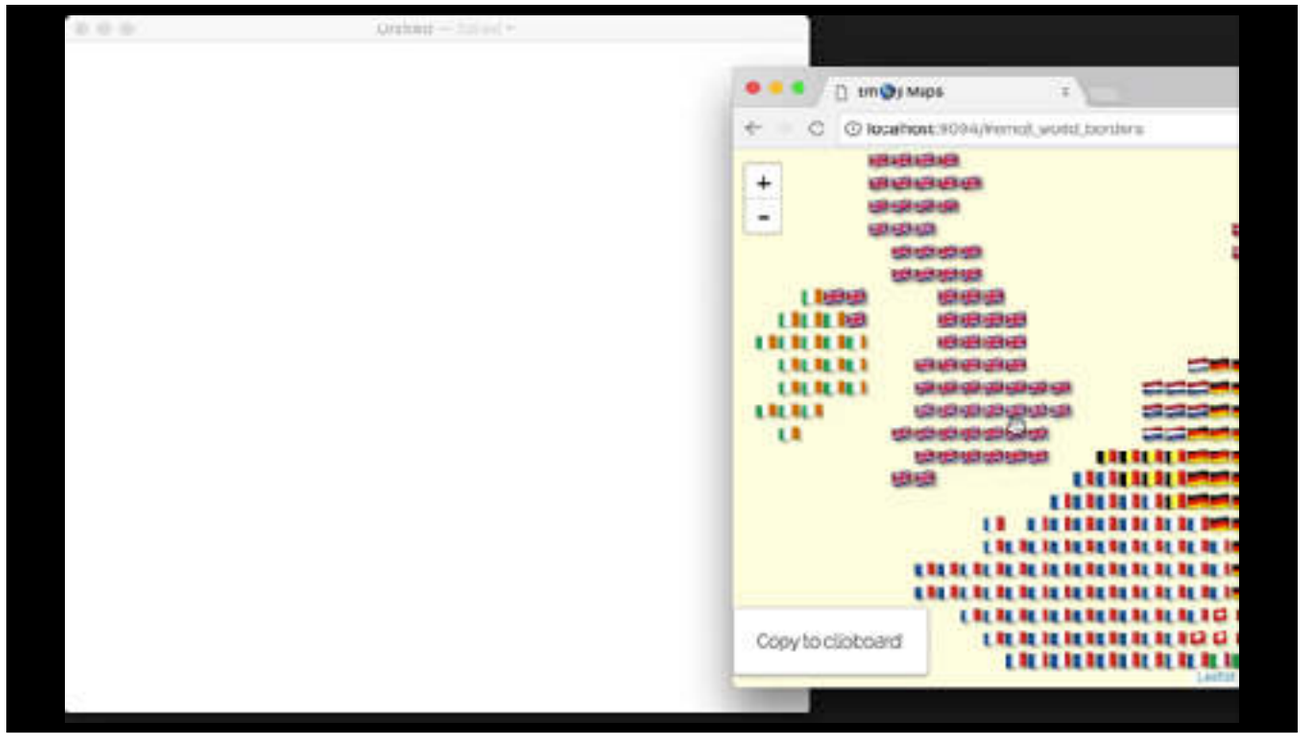
- A pannable and zoomable map made of emojis
- As a 🍃 plugin, named Leaflet.Emoji
- Renders GeoJSON polygons into a grid of characters, into a single textarea

Thematic Mapping With Emojis

My proposition for this talk is to unravel this experiment and discuss a few things I learned along the way.

A copy-pastable map

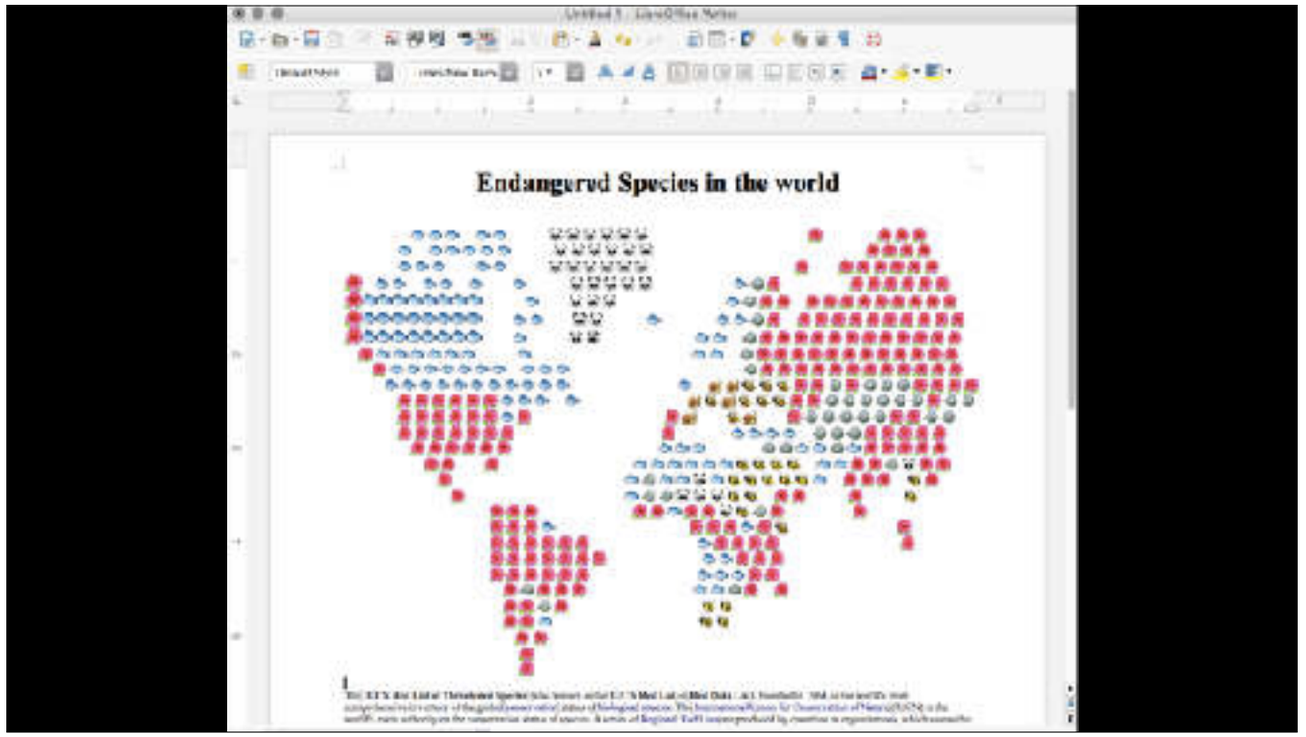
This is I believe the main proposal here. Let me explain this.



Because the map is rendered into a single string of text, you can easily copy it, paste it, and manipulate it at will, with just a keyboard.

GIS for the “emoji natives”

So we all now how making maps on a computer, or GIS, is a notoriously nebulous for outsiders. I believe allowing people to use only a keyboard to edit maps is potentially a powerful idea.



Especially when considering targeting a demographic that we can call ”emoji natives”, you now maybe that’s a way to talk geography with gen-Zers? I’m very intrigued by this idea.

A few things you need to know
about emojis 🤓

But at some point we need to stop and discuss about this: what are emojis?

Why do we have:

But no:
Gazpacho
Sauerkraut
Boeuf bourguignon
?

Who are the people
in charge of this ?

THE

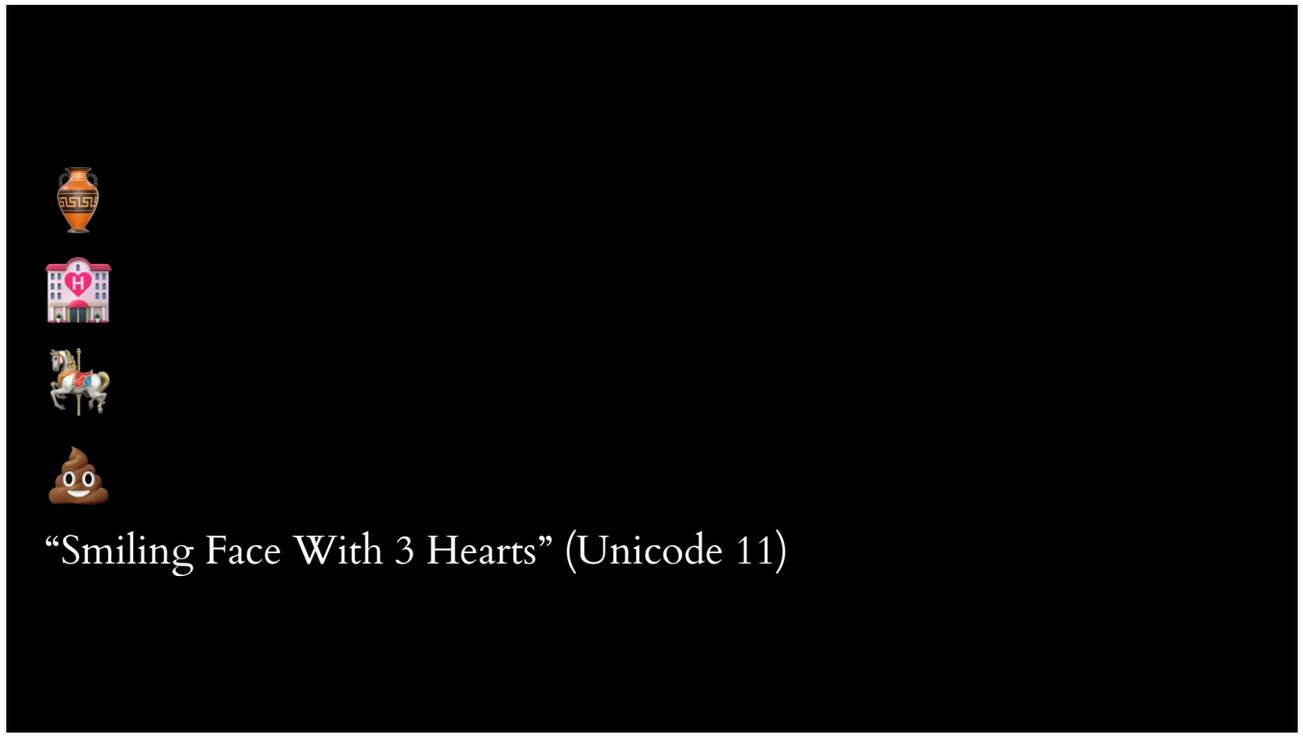
THE UNICODE

THE
UNICODE
CONSORTIUM

THE
UNICODE
CONSORTIUM
EMOJI

THE
UNICODE
CONSORTIUM
EMOJI
SUBCOMITTEE

(can’t help but imagine them like this)



And those folks get to decide that amphora, love hotel, carousel horse, pile of poo and “smiling face with 3 hearts” should be part of the Unicode standard, and therefore exist into this whole emoji subculture.

The Unicode Consortium coordinates the Unicode standard. Members are the like of Apple, Adobe, Facebook, Google, etc (voting members are exclusively American or asian, hence the gastronomy bias seen earlier)

The consortium’s mission is to encode in a standard every character ever produced by humans. Here a candidate proposal for medieval punctuation signs.

Which could seem a bit obscure, but it’s their work on emojis that focuses most of the public’s attention (including, yes, Stephen Colbert).

Anatomy of an emoji

- Unicode name
  “Hatching chick”
- Unicode codepoint(s)
  U+1F425 / 128036
- Shortcode
  :hatching_chick:
- Rendered glyph
  

So an emoji is made of what?
The final rendered glyph depends on the software, the platform, but also the font used.
Recommanded reading: https://www.smashingmagazine.com/2016/11/character-sets-encoding-emoji/

Emojis can be a combination of multiple code points

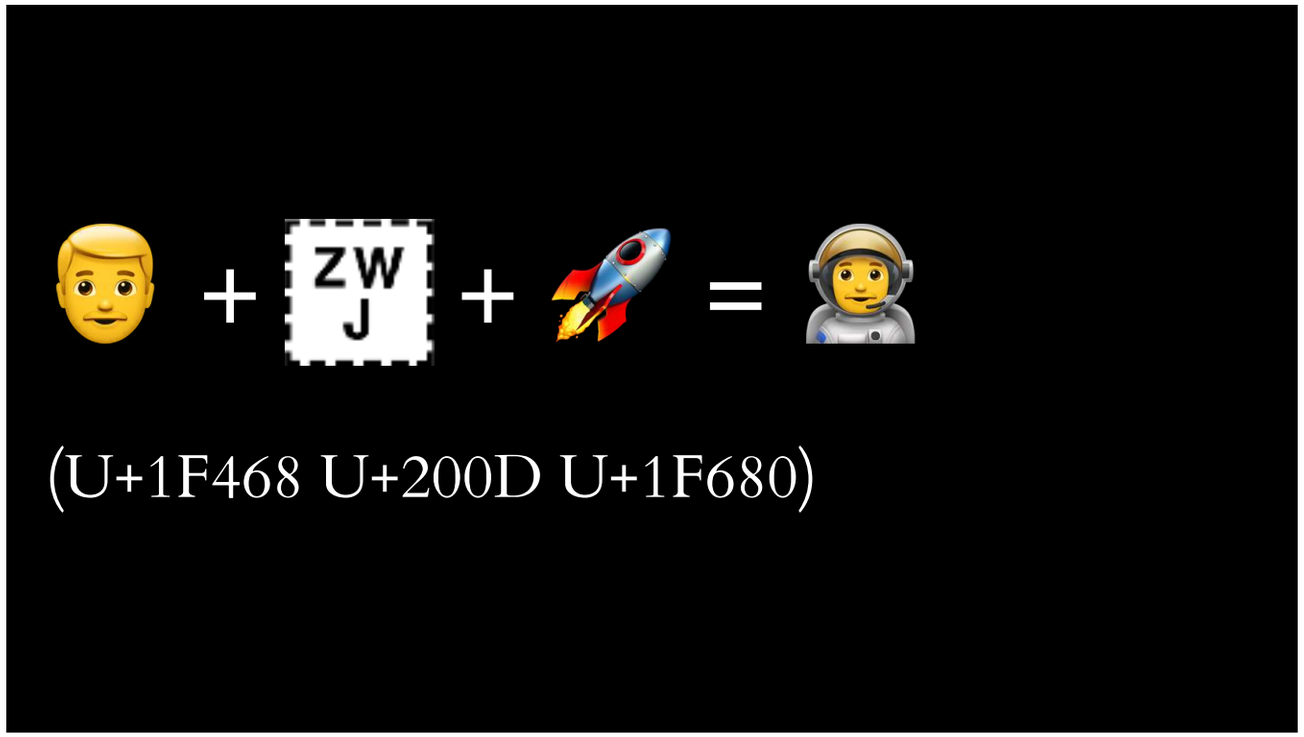
That’s an interesting and smart Unicode feature.

“old woman” + skin modifier
(U+1F475 U+1F3FF)

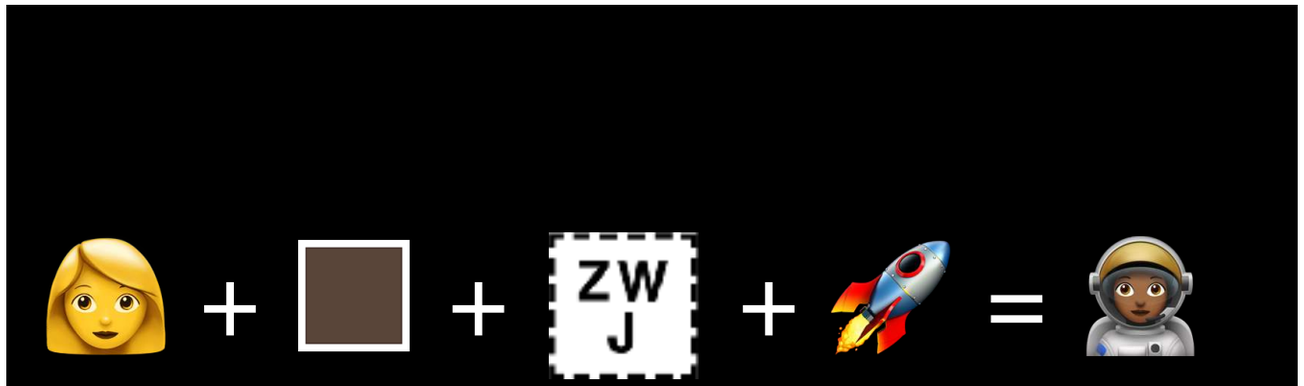
The Zero-Width Joiner (ZWJ)

A special, invisible character that allows us to combine multiple emojis into one

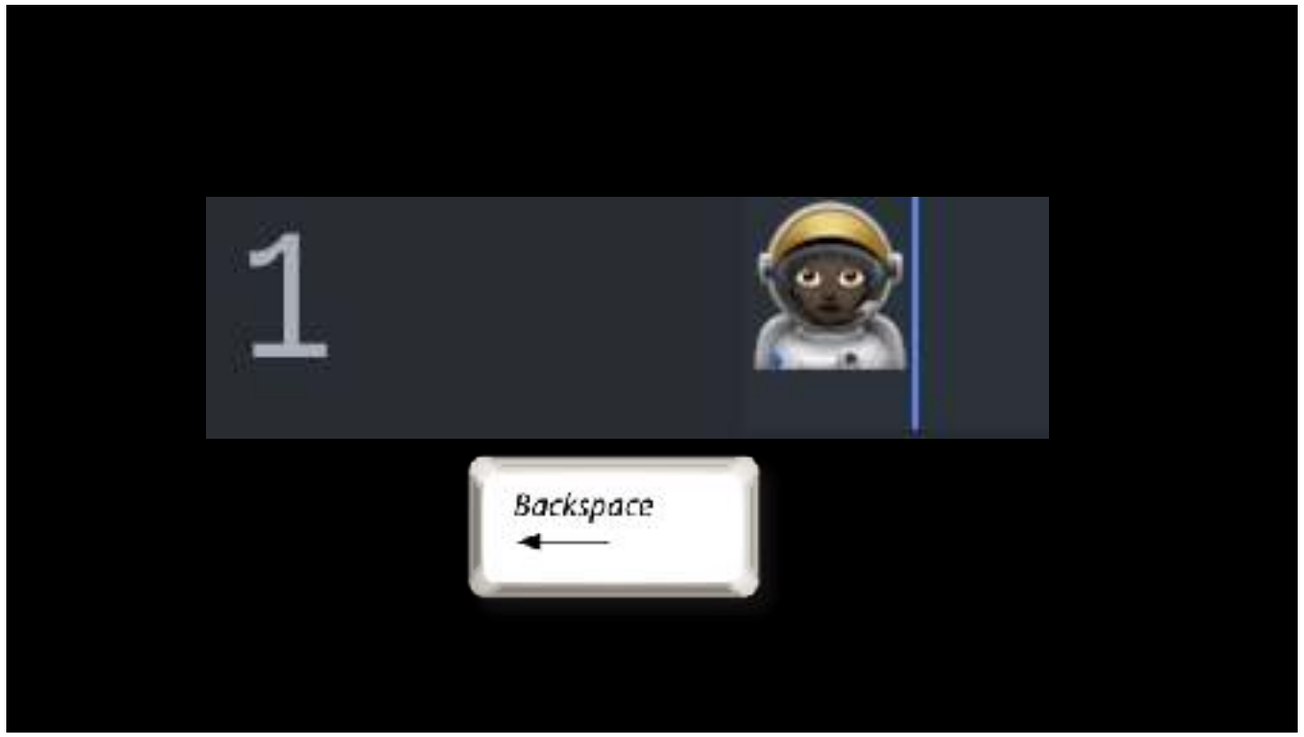
(Unicode: U+200D)



(U+1F468 U+200D U+1F680)



(U+1F469 U+1F3FF U+200D U+1F680)


1
Backspace

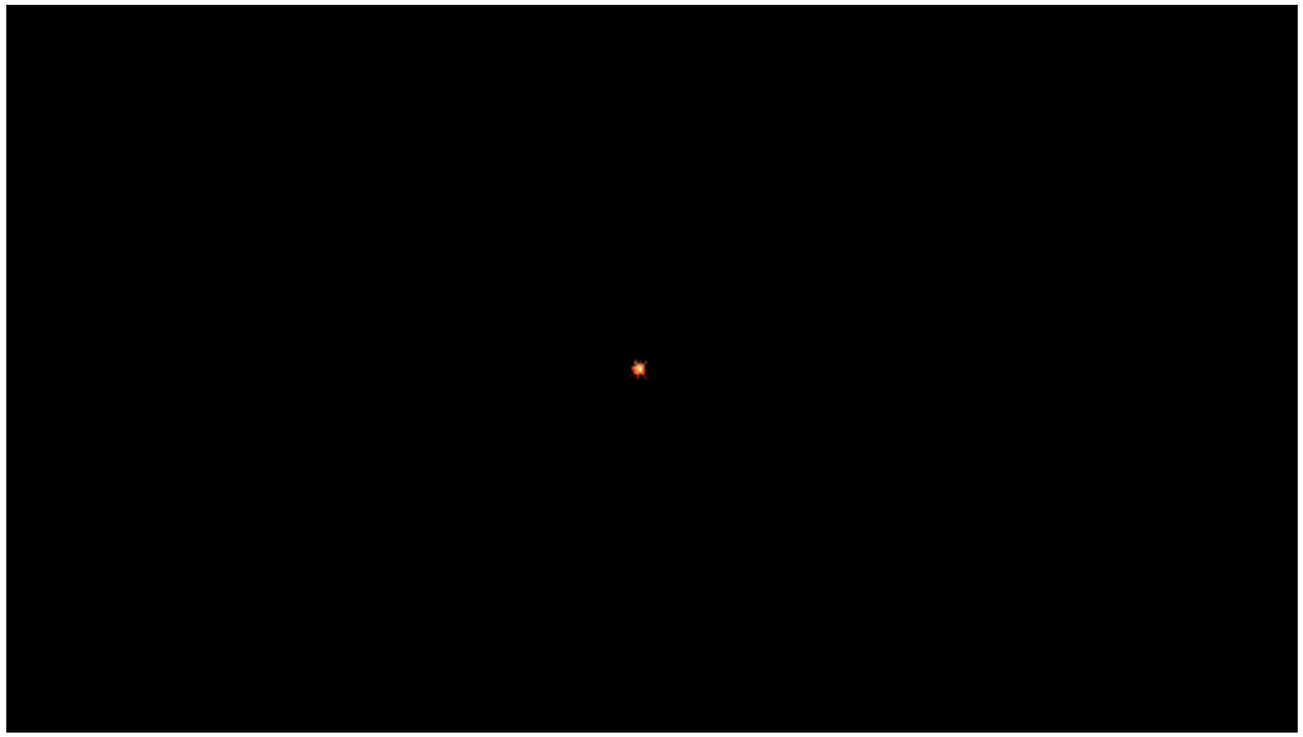
Emojis are designed
to add nuance to written
communication

Or sarcasm, double entendres, etc

But also:
A language of its own

What really interest us is a way of hacking around the designed use

Chronology of the universe:
Big bang, first light, earliest universe, formation of the solar system, sun, earth, apparition of water, photosynthesis, multicellular life, dinosaurs, mammals, humans, then the great human exodus from earth

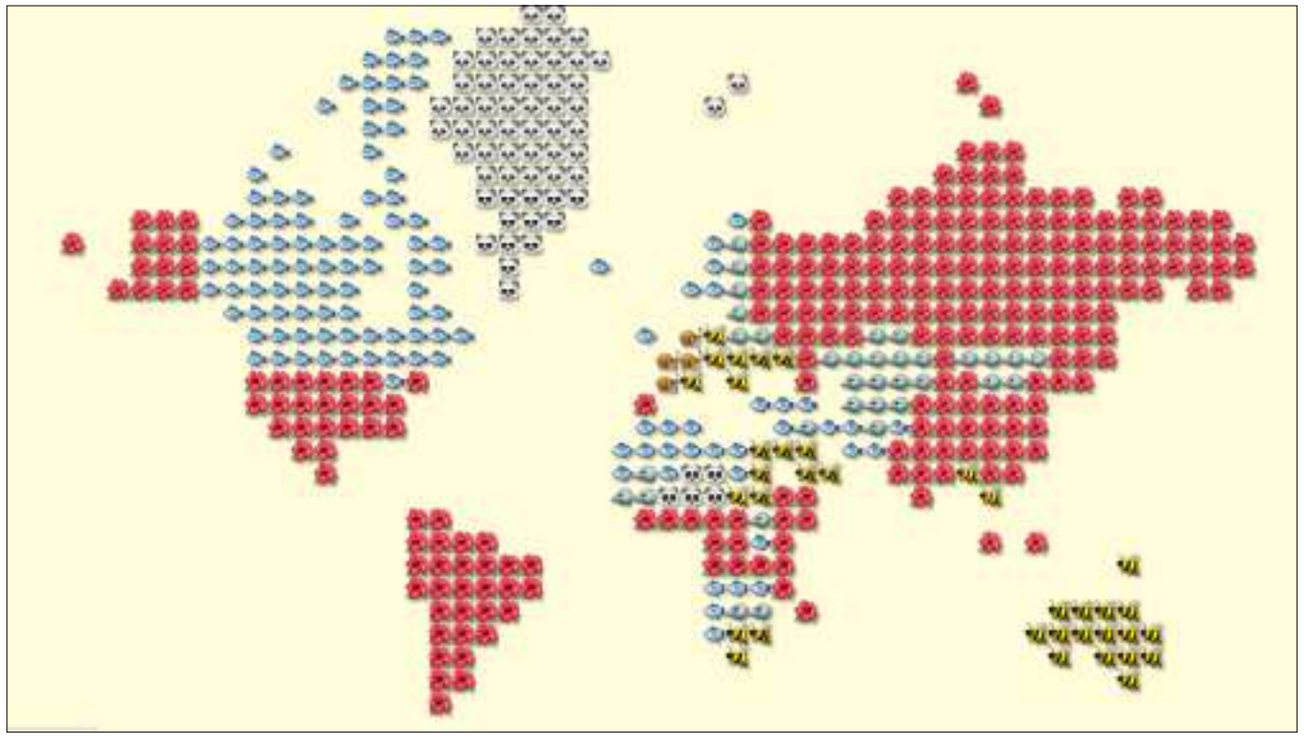


It’s a powerful way to express ideas and tell stories that allow for a lot of graphic design tricks

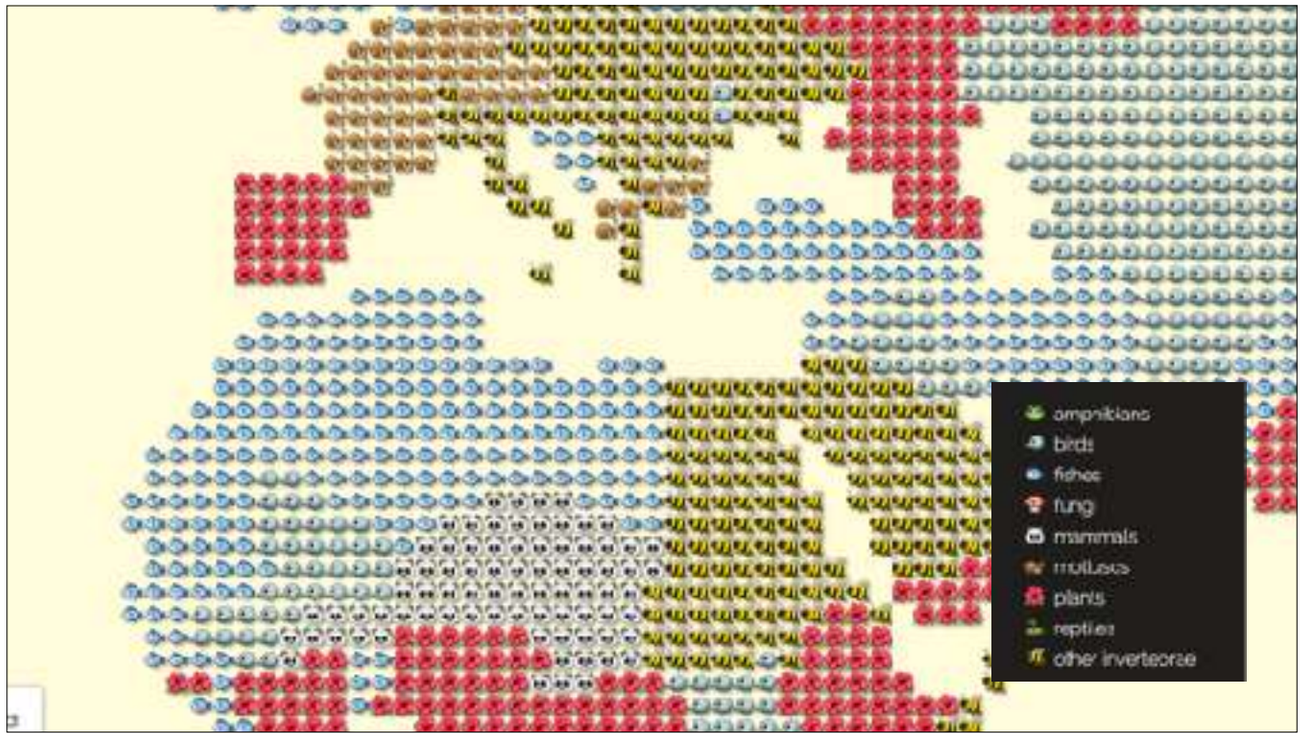
An experiment in
Thematic mapping

All of this makes great basic ingredients for great, expressive maps.

Proposition 1:
Qualitative maps

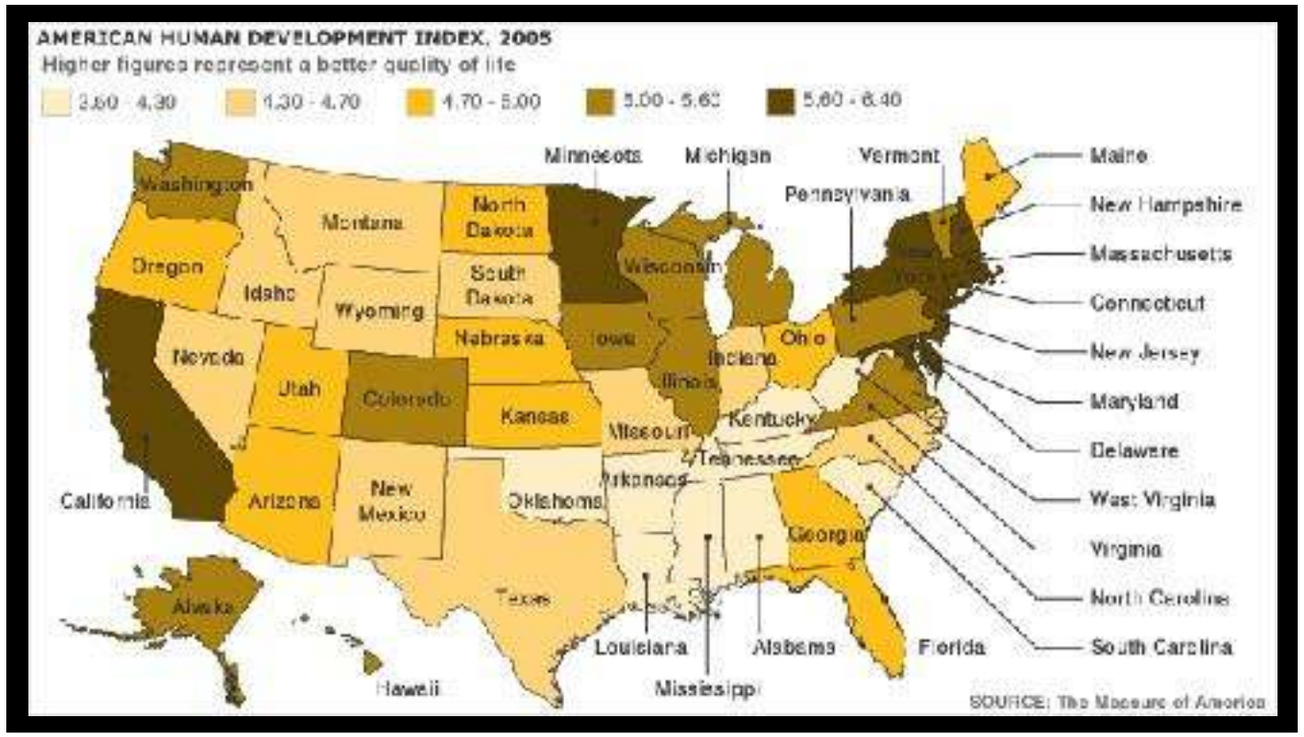


IUCN Red list of endangered species
Instead of mapping a value to a color, we simply map a value to an emoji.



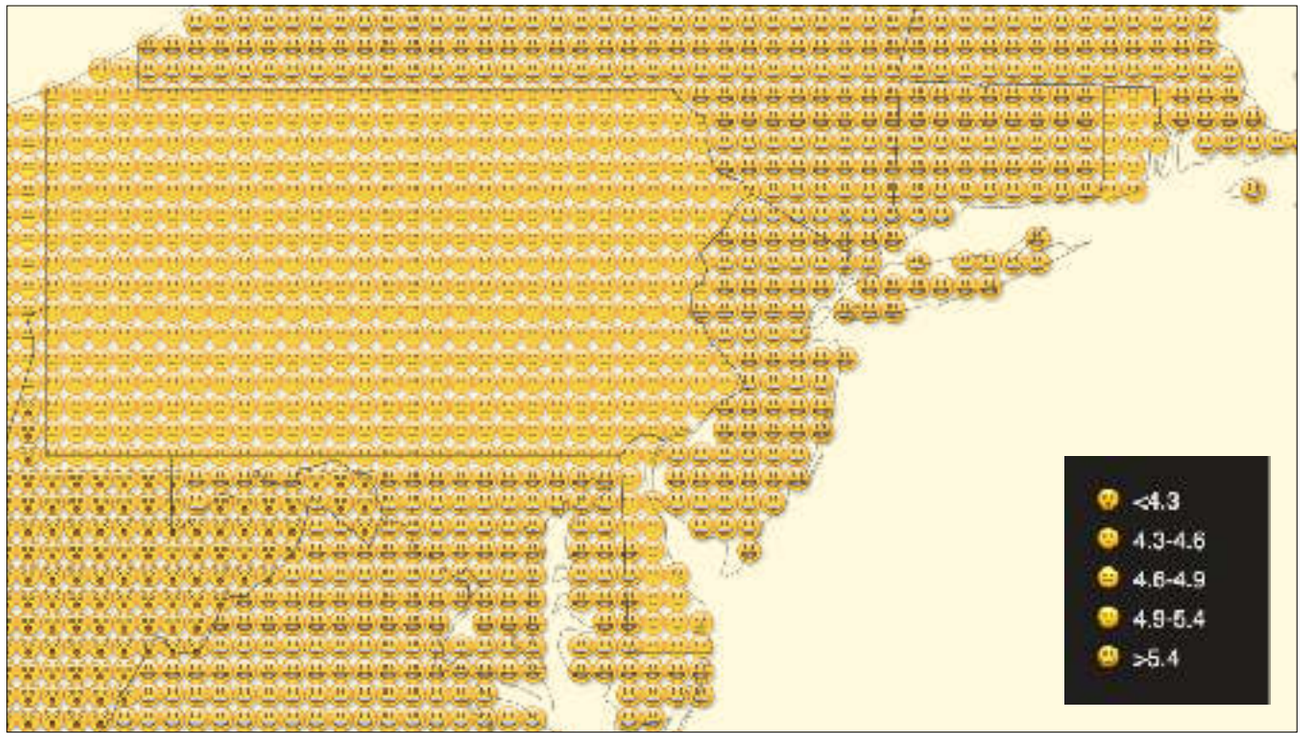
Here, we represent for each country the most endangered taxonomic group: mammals, insects, plants, etc

Proposition 2:
Emoji Choropleth

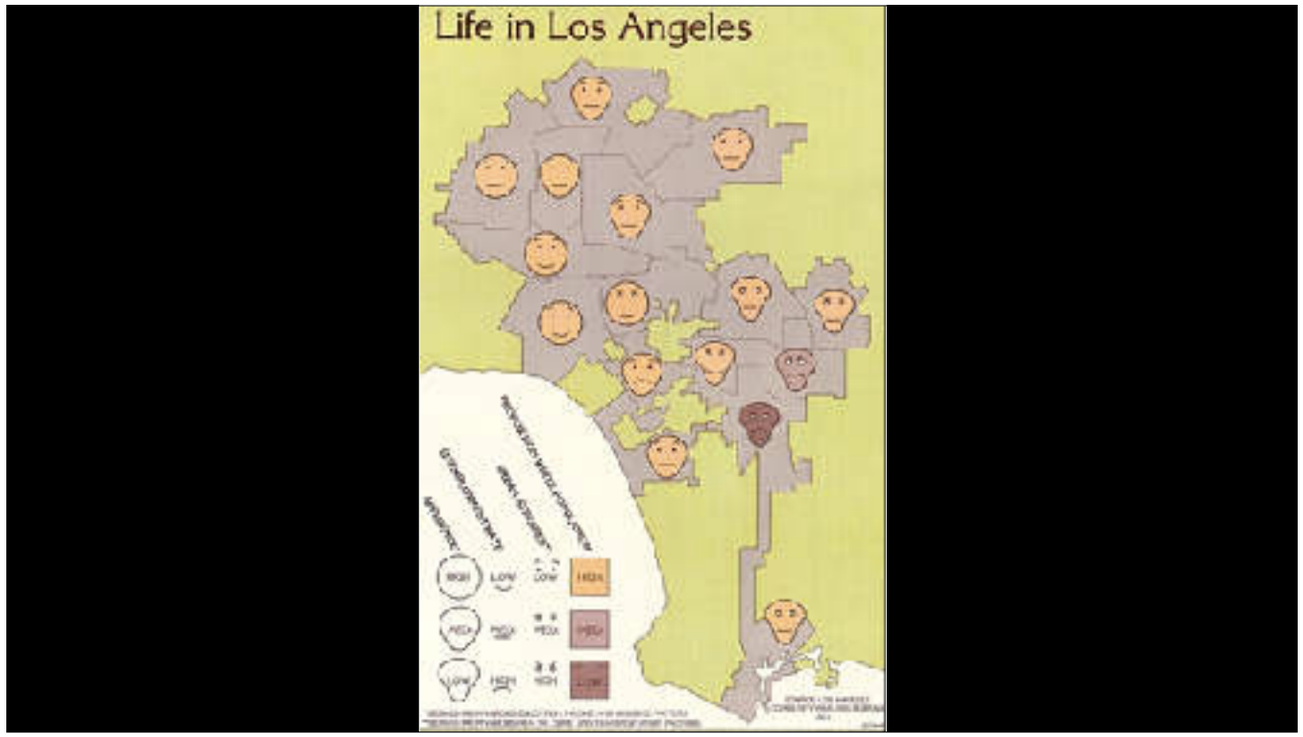


Let’s have a look at a choropleth. A statistical value gets encoded into a color. Here the American Human Development Index by US state (the HDI is a composite indicator to evaluate how a society is developed, here in its US “fork”, conducted by Measure of America)

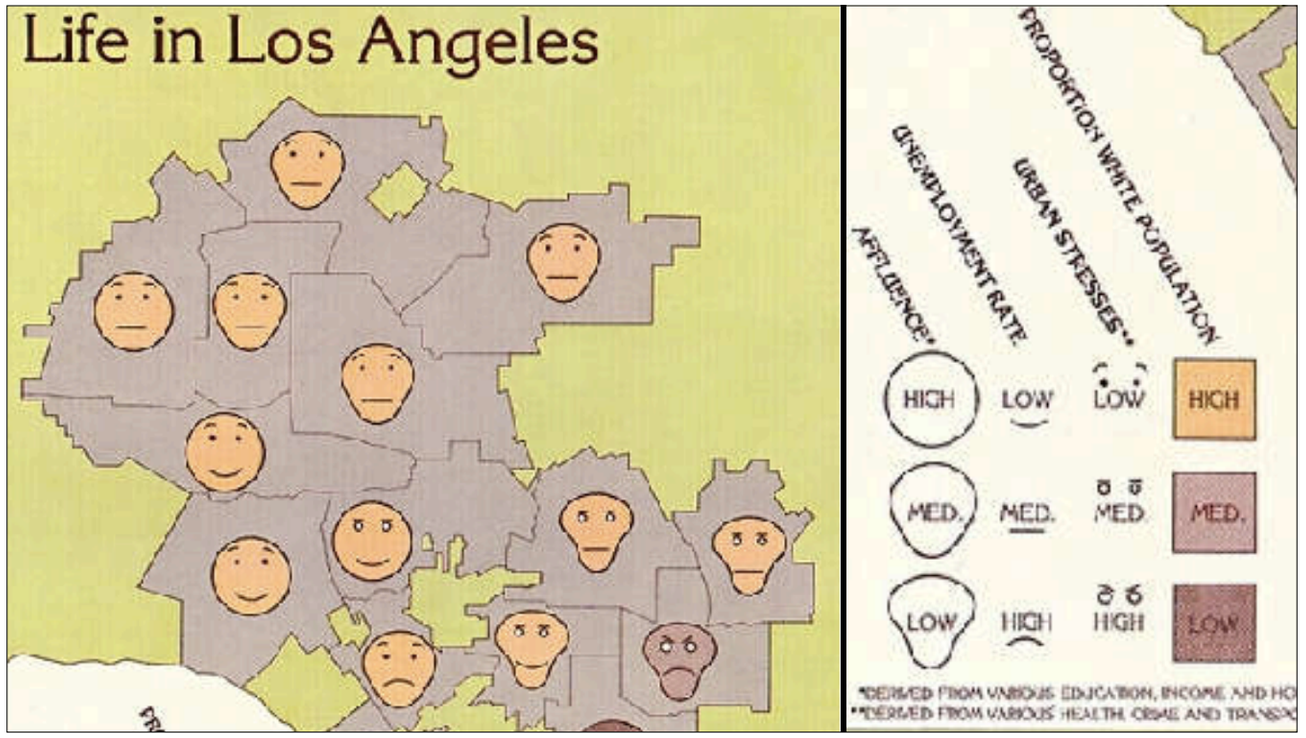
And this is the same map rendered using emojis, the iconic yellow faces.



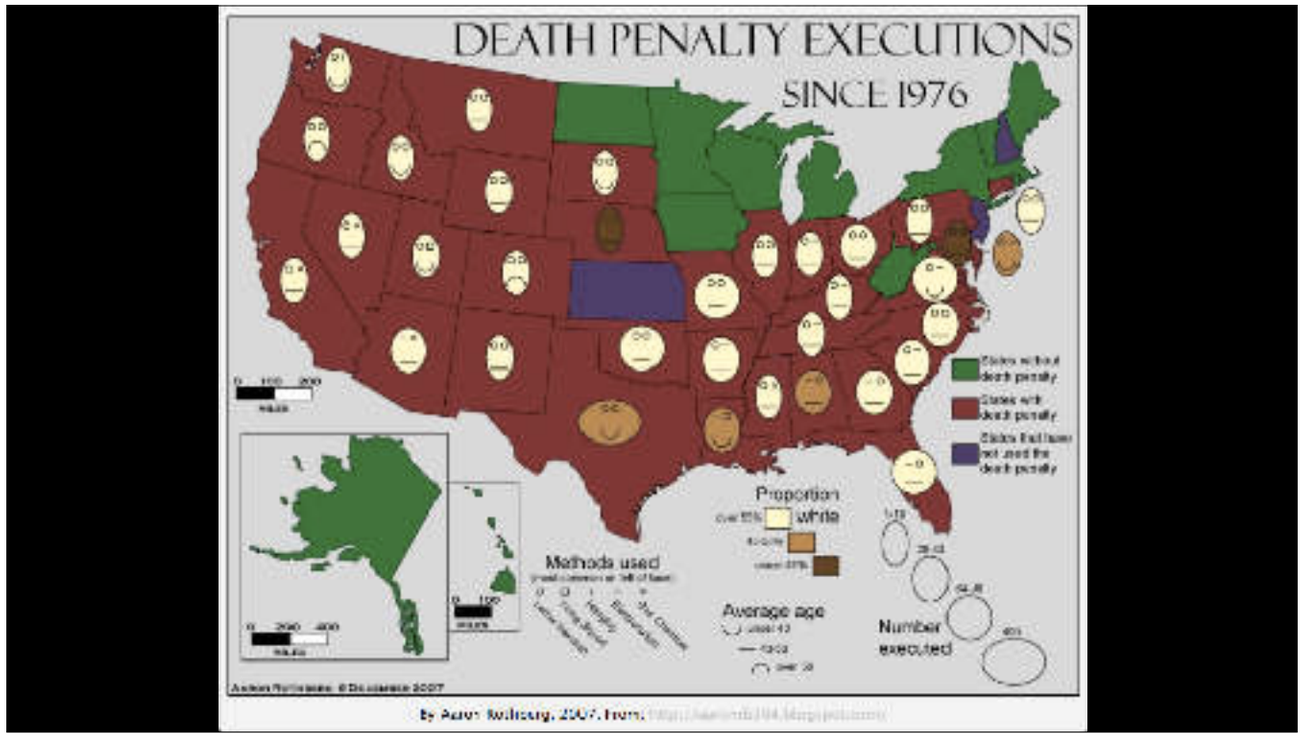
So we are simply using the expressiveness of human or emoji faces to render a value that has a negative or a positive connotation. This idea, to connect an indicator, preferably socioeconomic, to a human face, is not new…



In 1973, a statistician named Herman Chernoff proposed this idea of symbolizing data using human faces. It seems it’s a information design classic, personally I learned about Chernoff when I started digging this topic and I find it fascinating.



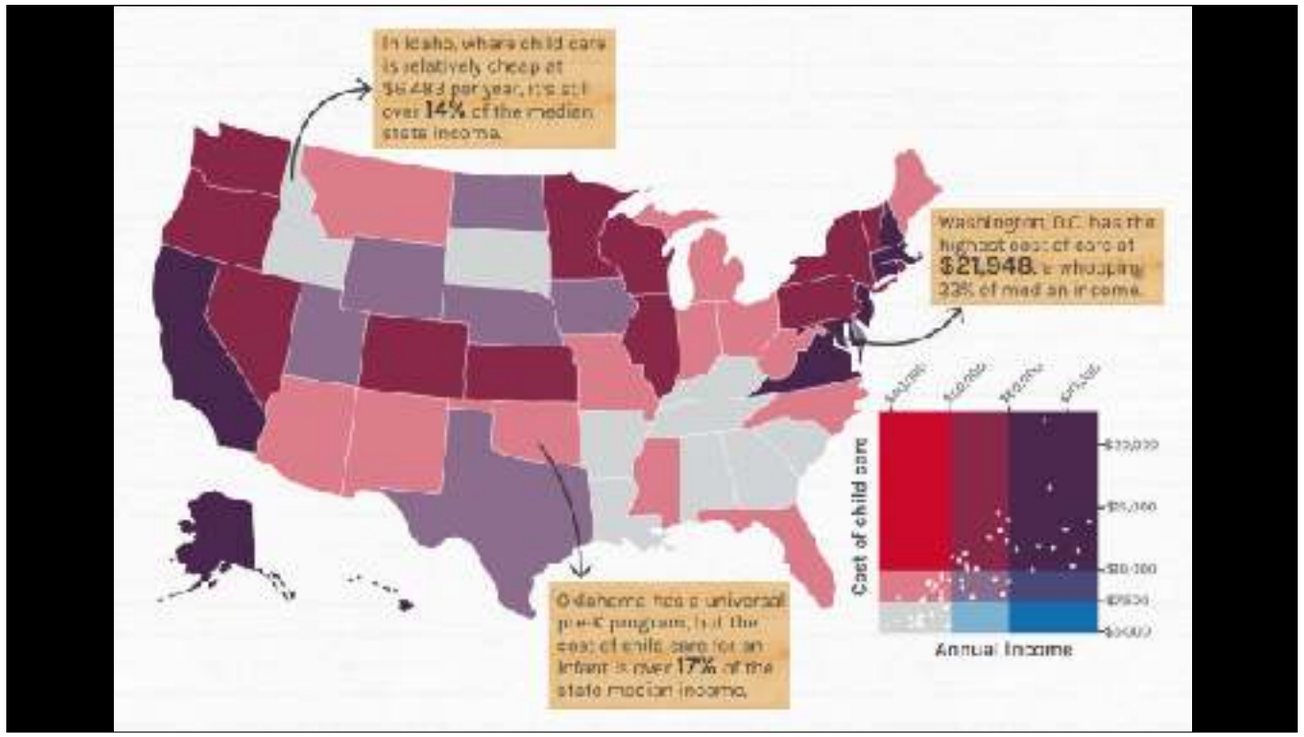
Chernoff’s idea was to encode several variables into a human face: shape of the head for affluence, shape of the mouth for unemployment rate, eyes for urban stress, face color for the proportion of white people, etc.
This supposedly, leverages our brains ability to read human faces extremely quickly and easily.



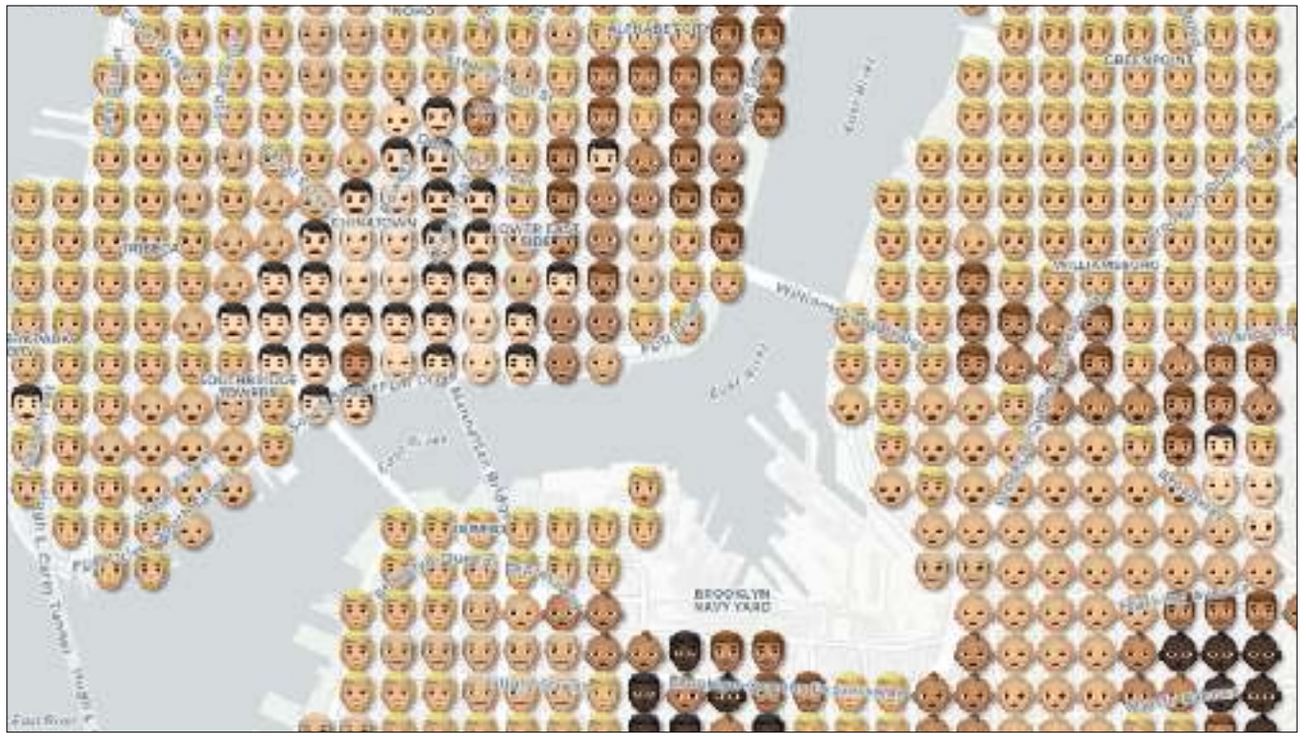
Another example that pushes this theory to the extreme. Here a large smiling face with round eyes means: in this state, more than 400 people of under 40 on average have been executed with lethal injection. One could start seeing the problem with this technique.

Chernoff Faces is the kind of information graphics that the only the 70s could produce, I guess. Maybe this idea of using emotions to encode cold statistics deserves to be kept though? This kind of works here because there's a universal agreement that a low value is bad and a high value is good.

Proposition 3:
Emoji Multivariate Choropleth

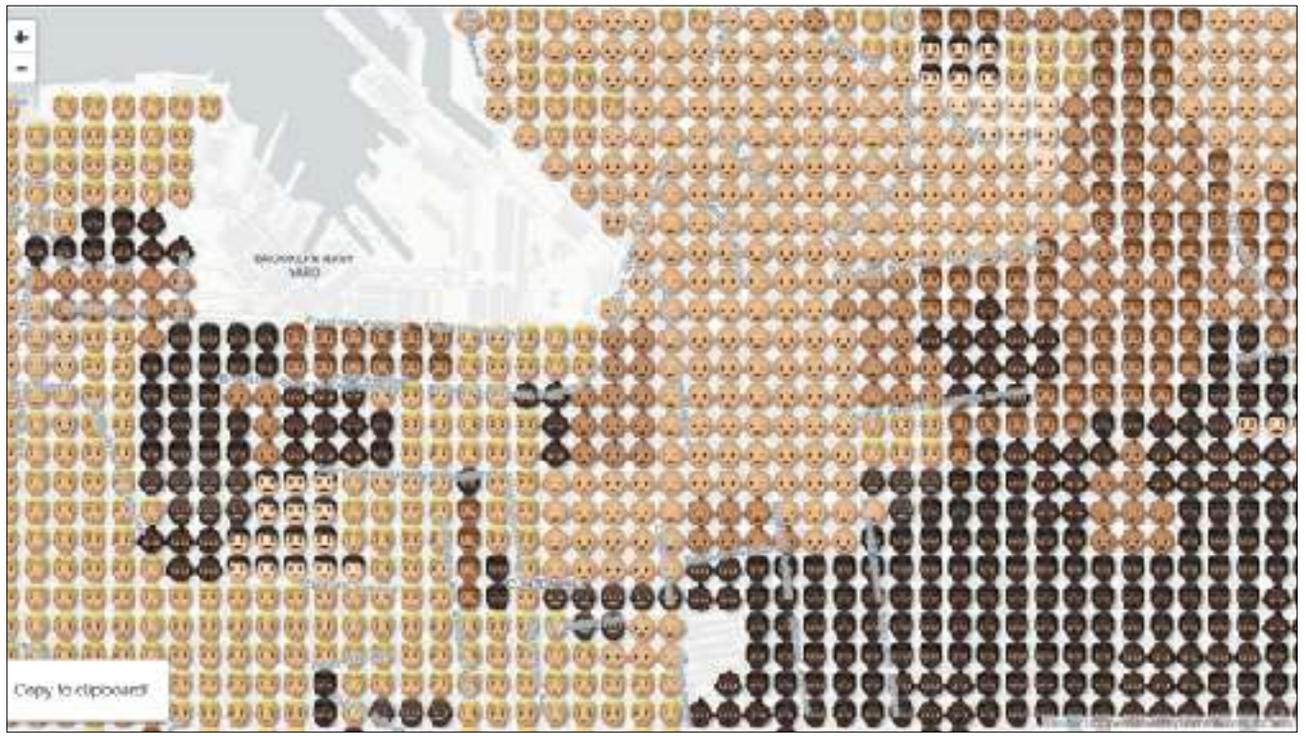


A choropleth can encode more than one value. For instance this bivariate (bi as in two variables) tells us, for each state, what is the median state income AND the cost of child care. This is technique that has the advantage of highlighting outliers, but has the disadvantage of requiring some work for the reader to make sense out of it.



So here’s a proposal: encoding two variables with emojis. Here, we see both median age and predominant ethnicity for each census tract in NY.


NEW YORK
Copy to clipboard



Basically this gives us an idea of “what is the most likely type of human I am to encounter in a given area”

median age:

- <30
- 30-45
- >45

predominant ethnic group:

- asian
- caucasian
- hispanic or latino
- african american

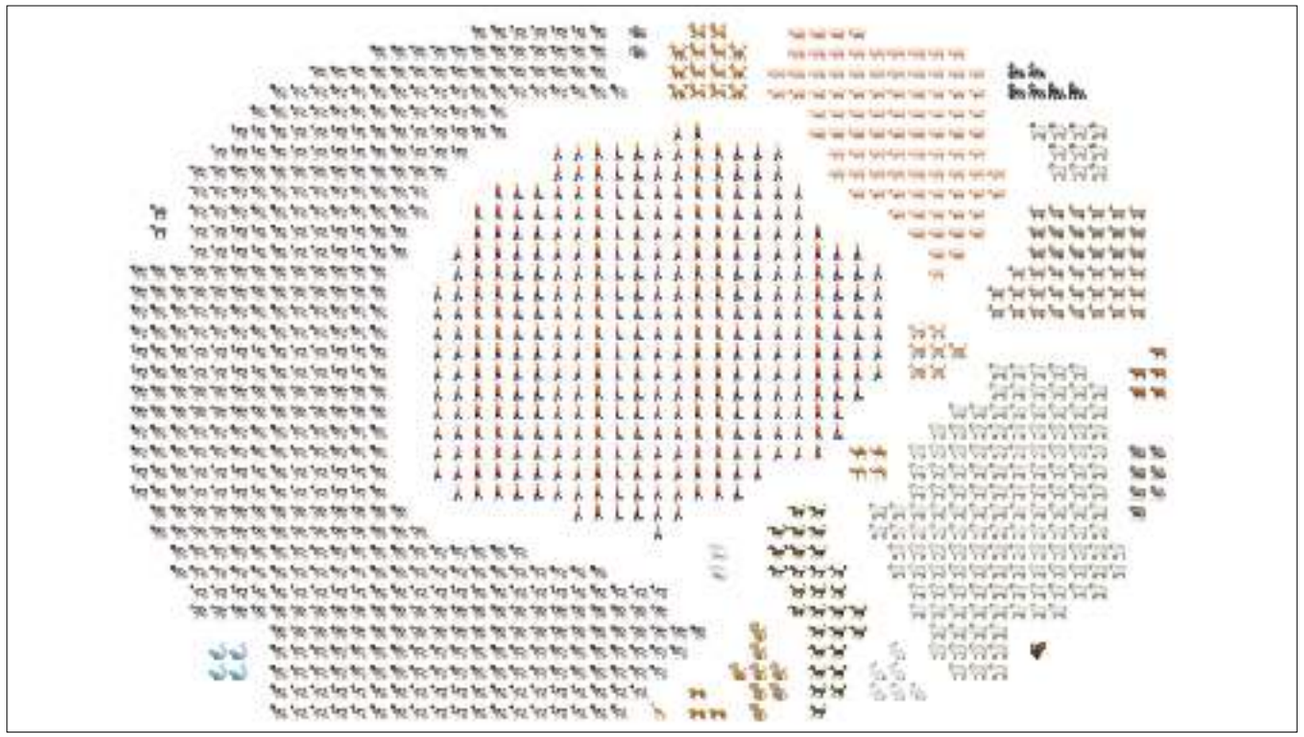
summary

<35	30-45	>45	
			asian
			caucasian
			hispanic or latino
			african american

There’s meaning even in the code.

Proposition 4:
Emoji Infographics

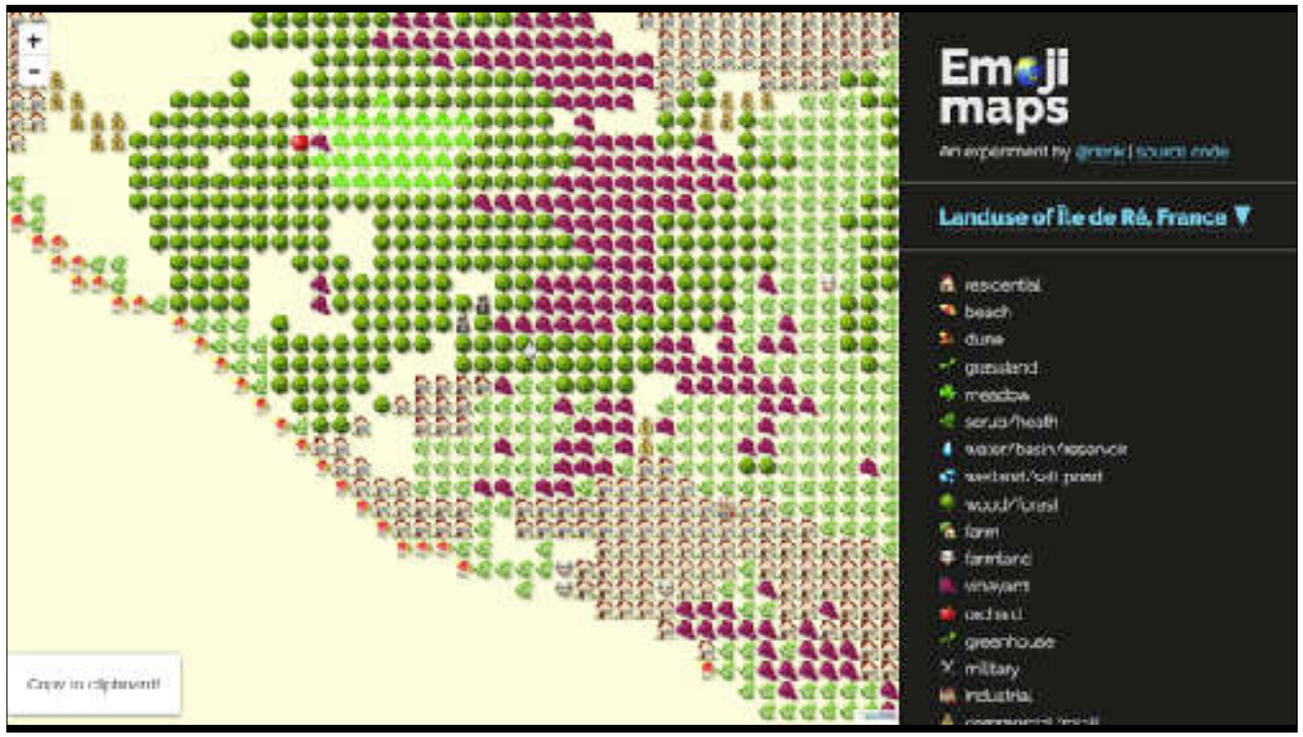
Beyond maps, emoji can be used in other type of infographics.



Many people have actually already done that, but this is our take on it: global animal biomass on earth

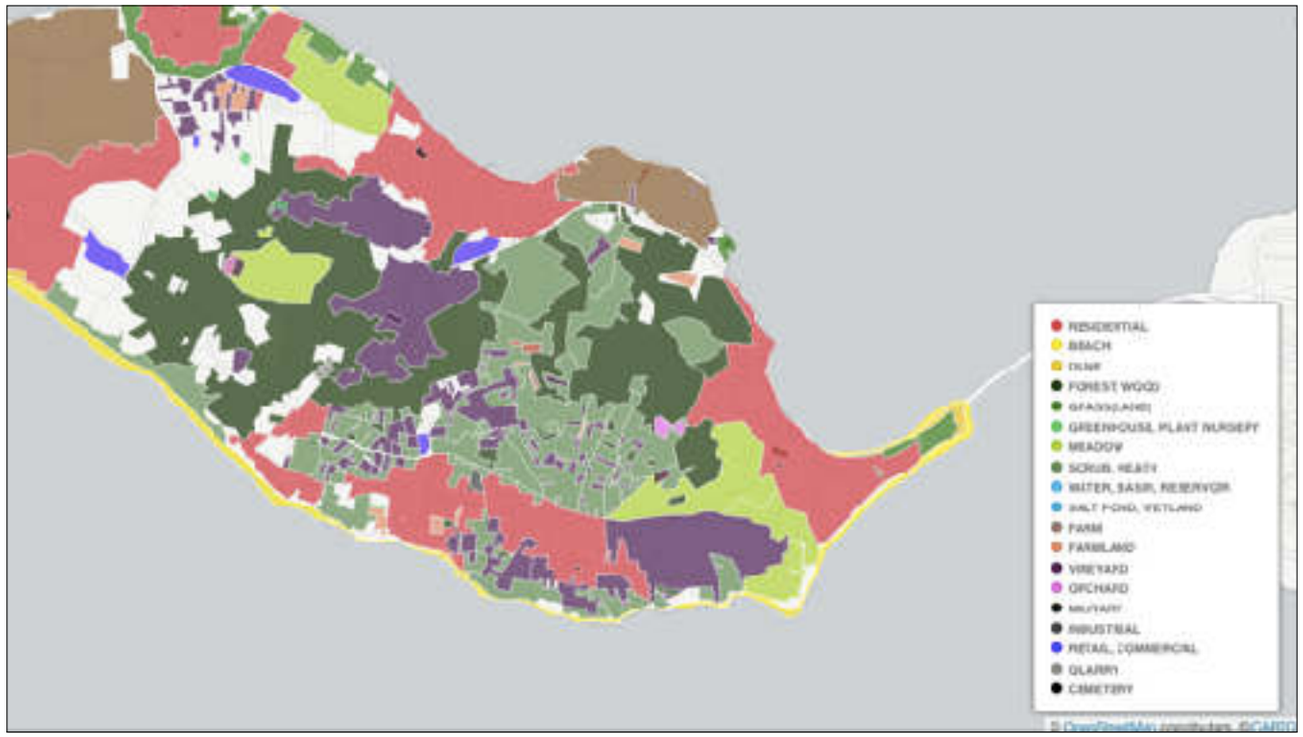
Proposition 5:
Getting rid of legends

A common trait in all the previous propositions is that we make map legends much less necessary because that pictogram grid is kind of self-explanatory.

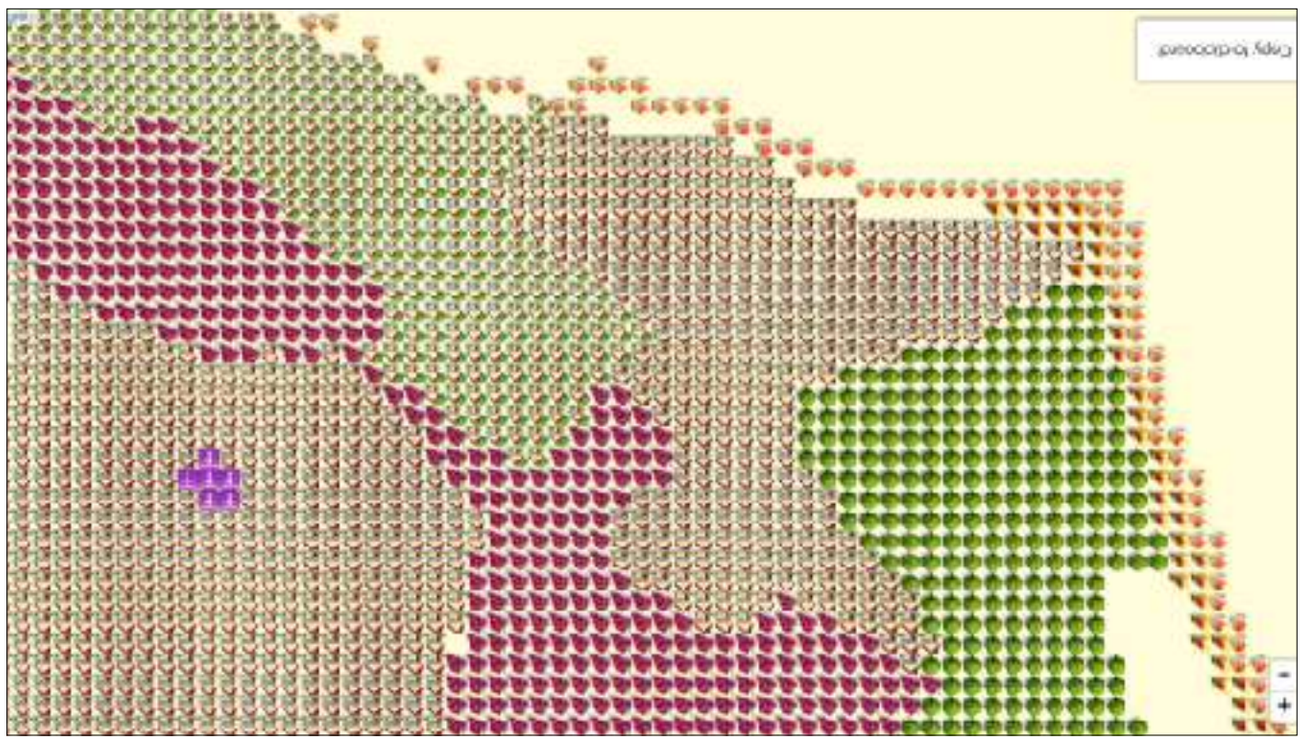


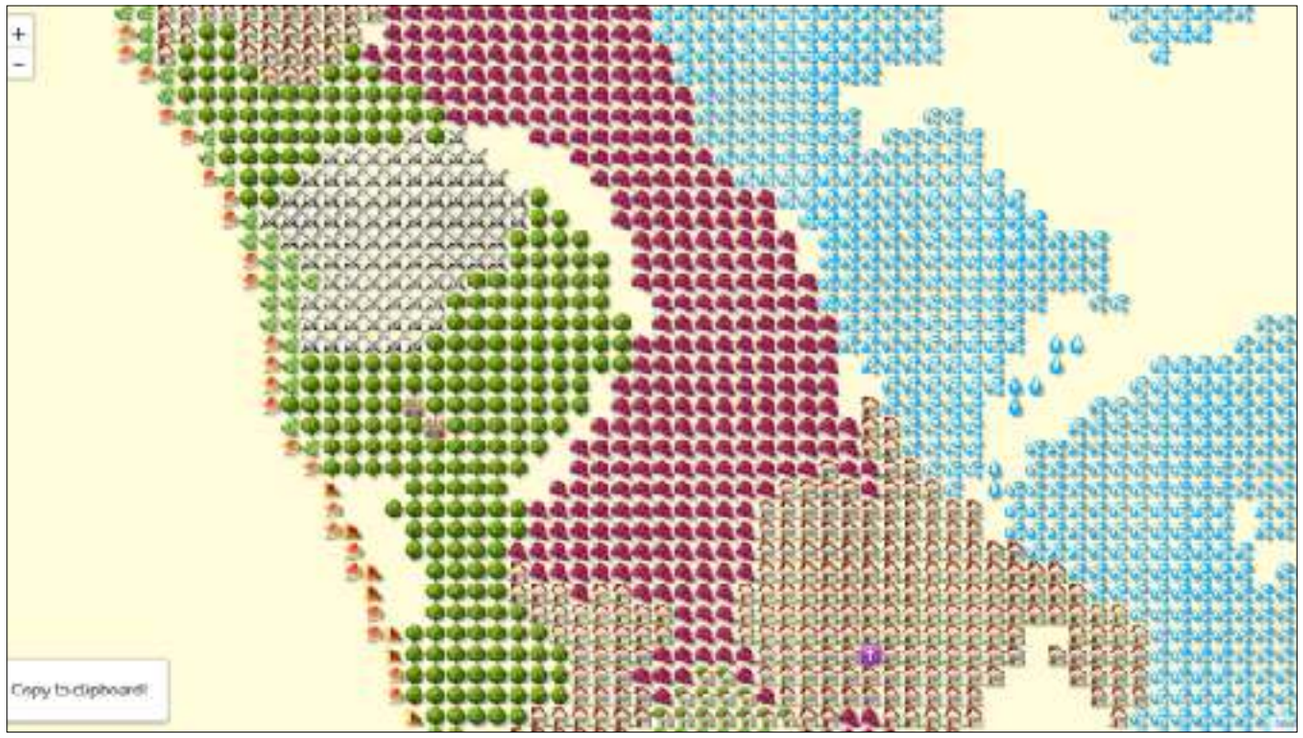
This is even more true for a dataset with a lot of classes such as mapping landuse, ie dividing a territory by type of human use.

This is a the Ile de Ré, a beautiful island on the Atlantic coast of France.



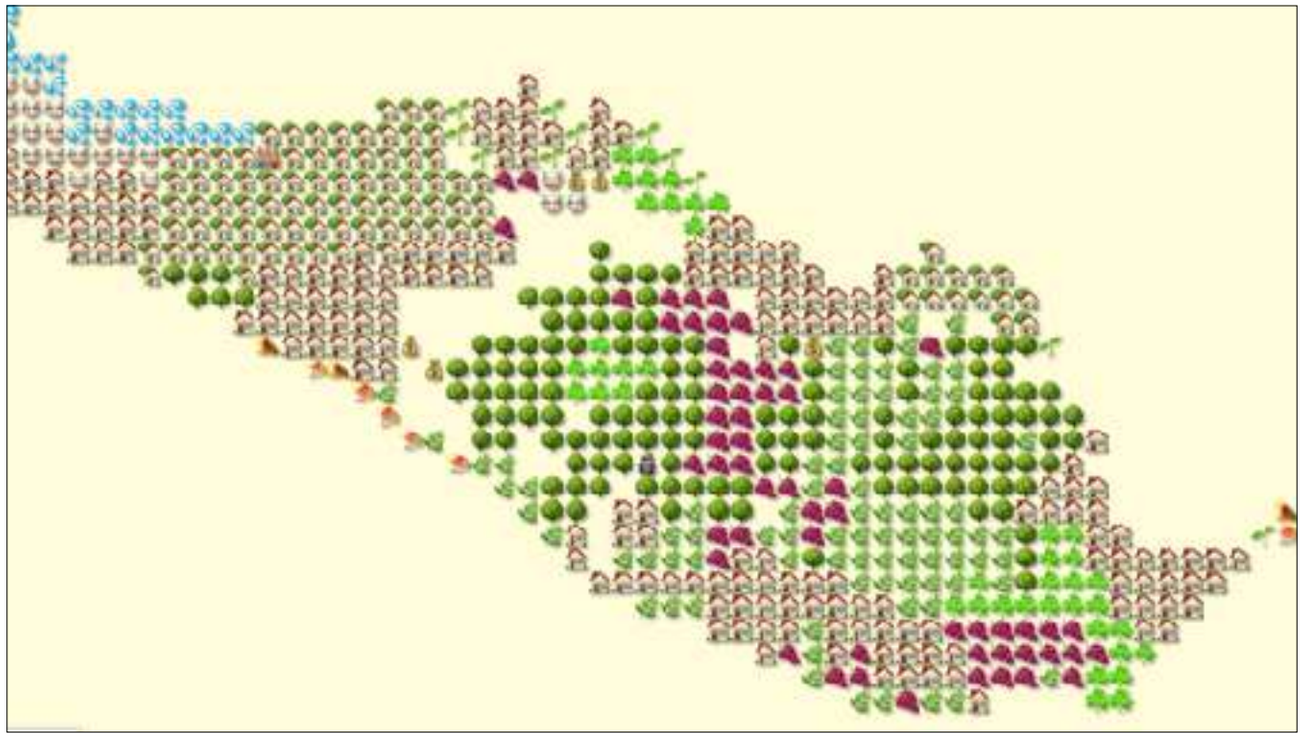
And a classical qualitative map would look like this. There are so many classes that there’s no way around adding a legend. (this, by the way, is data pulled from OSM through Overpass).




+
−
Copy to clipboard

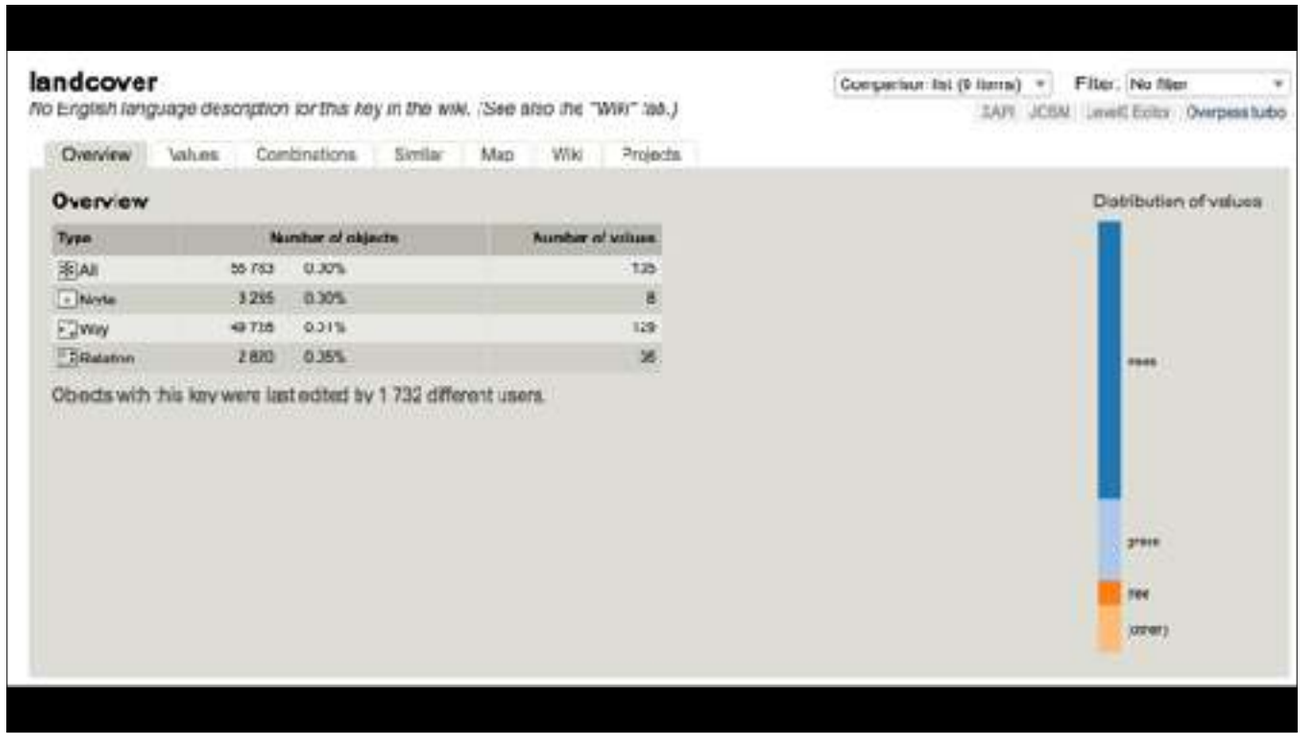
And even if we actually need a legend, to remove ambiguity, notice how more expressive it is.

The 🌍 in emojis with OSM



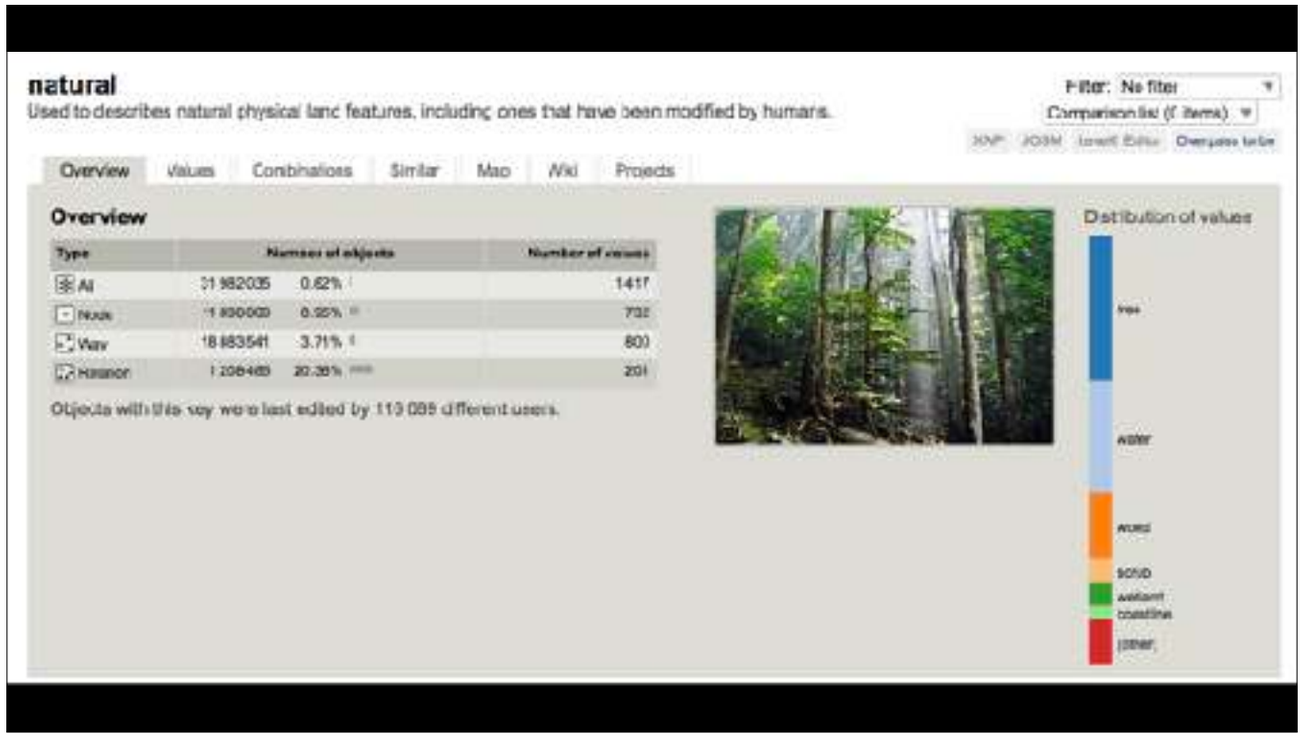
As mentioned earlier, this was done by using a JSON pulled from Overpass Turbo, but it’ll only work for that region obviously.

So this version is actually running a fresh, “live”, version of OSM landcover and landuse.



It’s very hard to make something that works on a small island in the French Atlantic coast, work globally. Where landuse was sufficient, we need to add to the mix landcover and water.

Some more could be used, but on “case by case basis·: man_made, natural, etc


natural
Used to describes natural physical land features, including ones that have been modified by humans.
Filter: No filter
Comparison list (0 items)
Overview
Values
Combinations
Similar
Map
Wiki
Projects
Overview
Type
Number of objects
Number of values
All
Node
Way
Relation
1417
732
803
201
Objects with this key were last edited by 113 089 different users.
Distribution of values
tree
water
wood
scrub
wetland
coastline
other

So what's the difference between
landcover=trees
landuse=forest
natural=wood
It's easy !!!

[Tagging] British term for municipal greenery?

Warin 61sundowner at gmail.com
Fri Jun 1 07:58:20 UTC 2018

- Next message: [Tagging] British term for municipal greenery?
- **Messages sorted by:** [date] [thread] [subject] [author]

On 01/06/18 07:20, Peter Elderson wrote:
> What would be a fitting term for municipal greenery? What I mean is
> areas ranging from a few square meters to say a 200 m², maybe more,
> shapes varying but certainly planned, with public maintenance, most of
> the time as decorative separator strips shielding objects from sight
> or passage, or just to fill up the land. We call that collectively
> "gemeentegroen" that is "municipal greenery", because the municipality
> owns the land and and has the flora maintained by the 'greenery
> service".
>
> Growth varies from just grass (as separator, not as park), large
> flower beds which are removed every year, bushes, low trees which
> no-one would dare to call a forest, and mixtures. Thy might be bushes
> one year, grass with flowers next year, and cactus-fields next year
> because the mayor has visited Mexico.

Assume you trying to tag the use of the land:

landuse=flowerbed/garden are both present.
However if the thing changes from time to time it may be best to use a
new value ... say landuse=decorative?

If you are trying to tag the land cover then

landcover=grass/flowers/shrubs/*

Note that this tag is poorly represented in the data base so it maybe
useful to dual tag with the misused tag natural=*

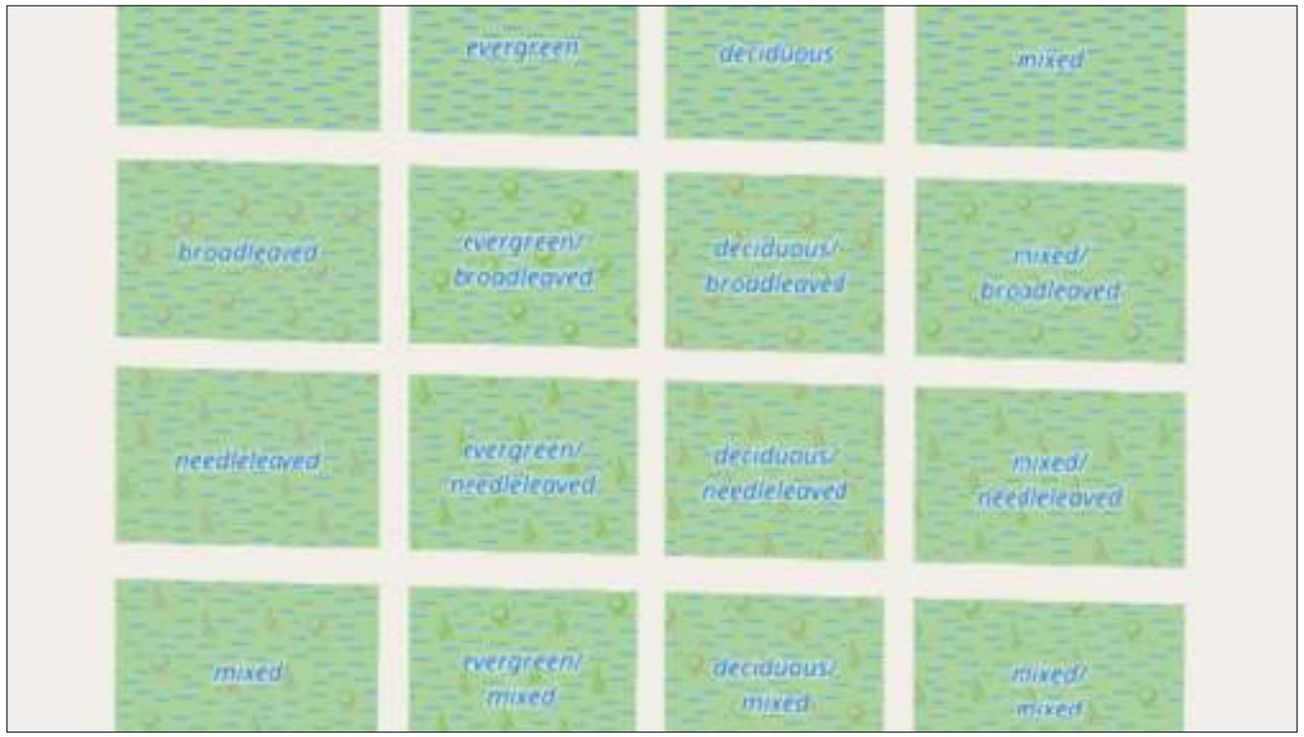
In either case you can add sub tag for operator=* .. eg name of
council/municipality/shire

Here someone asking on the ML what tag should be used for “municipal greenery”. The author of the answer here suggests 3 distinct tagging schemes…

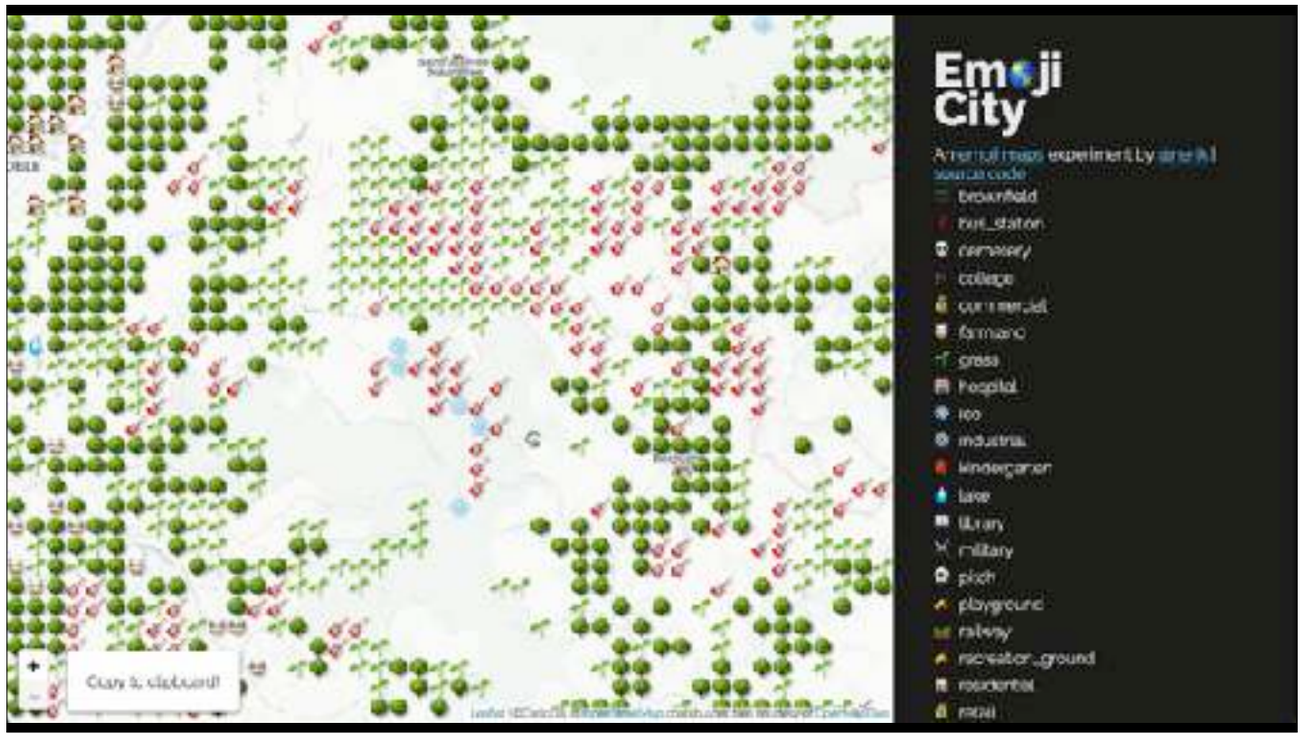
And ofc the conversation goes on.

Global map rendering decisions

I’m pretty much an outsider to OSM, so not going into any tagging debate, BUT: it’s a very instructive experience, at my very small, anecdotical level. I’m talking about making decisions for a planet-wide map style.



I very much recommend reading Christoph Hormann blog about that topic, in particular this epic post about rendering woodland. http://blog.imagico.de/differentiated-rendering-of-woodland-in-maps/

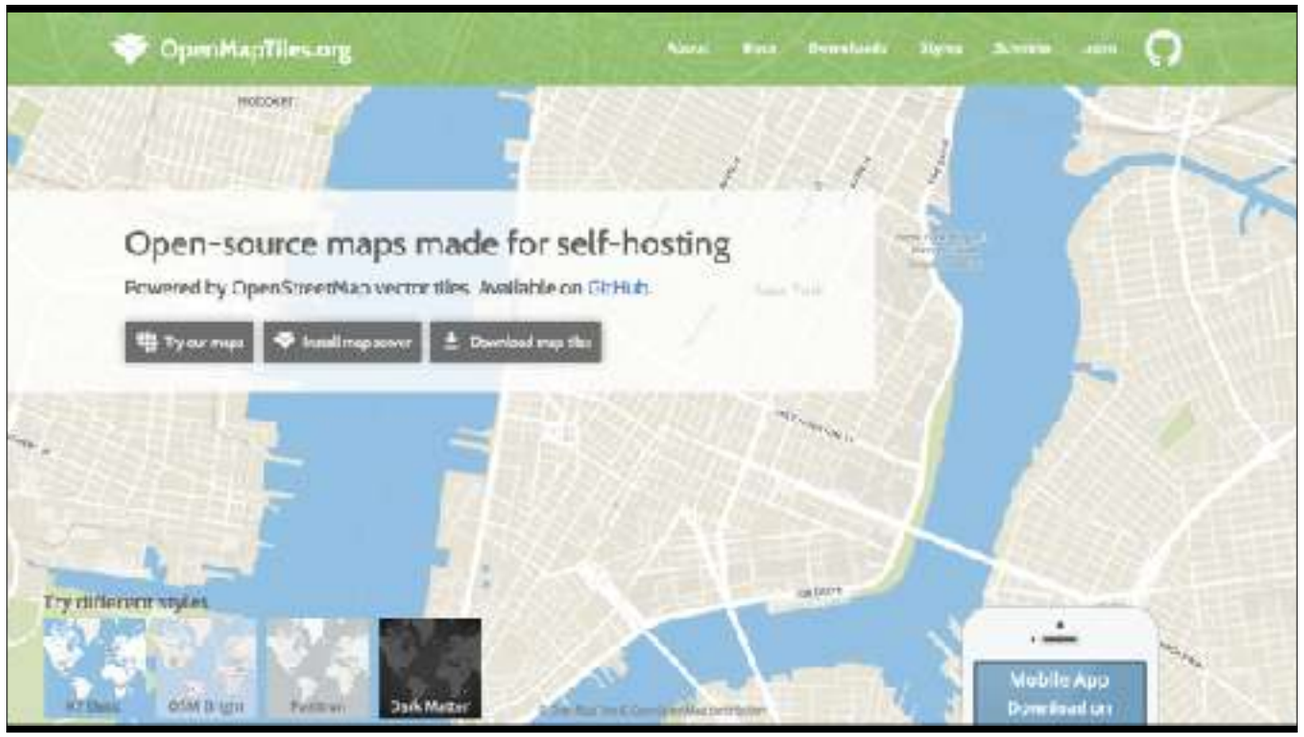


So yes natural=rock is represented as small electric guitars, because I can. It’s interesting to be in that seat.

An experiment with vector tiles: Exploring alternatives

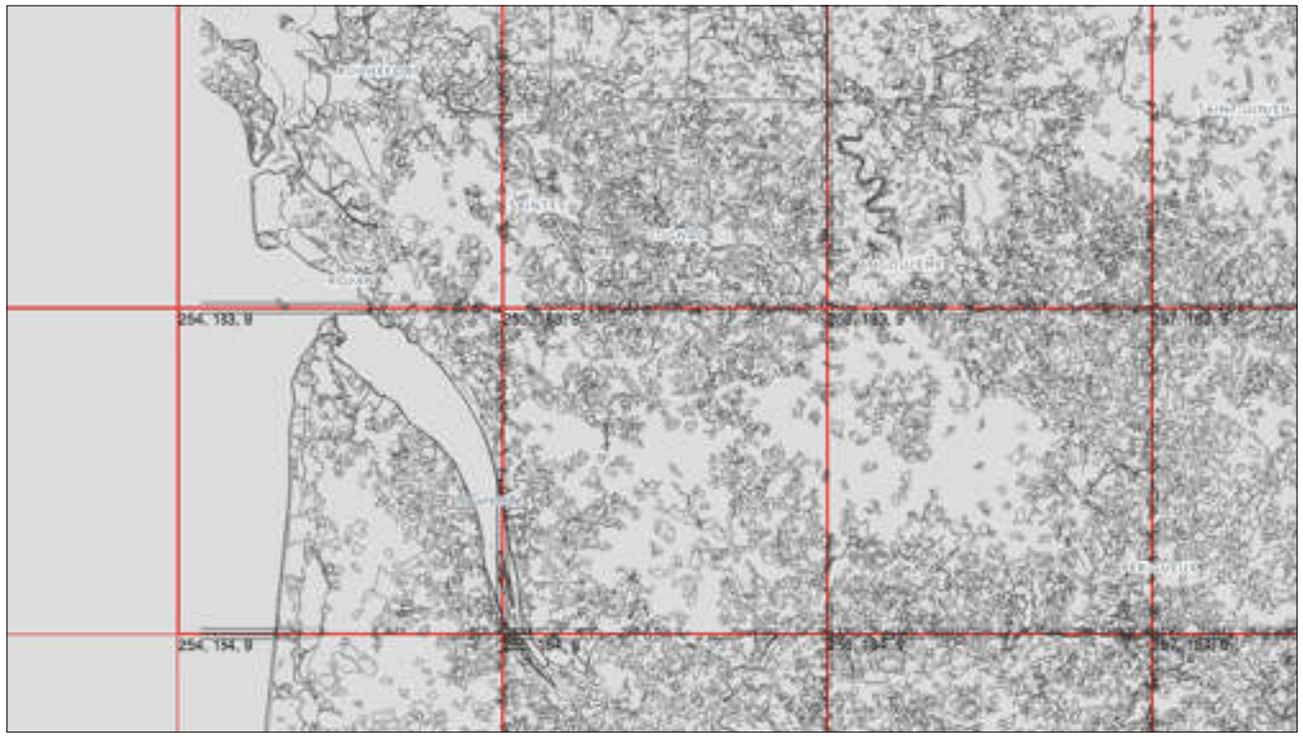
On a more technical note; We love the Mapbox stack (Mapbox OSM vector tiles + Mapbox GL JS) and we are using it all the time in our work. But we took this as an opportunity to explore different options.

We were using L from the start of this project, which was perfect for its versatility and simplicity. And also, Leaflet is incredibly well designed for extensibility.



Vector tiles are courtesy of OpenMapTiles.org, a refreshing alternative to you-know-who.

While Leaflet is not “vector tile ready” in the same sense than Mapbox GL is, it’s perfectly possible to load and read vector tiles with a standalone library.

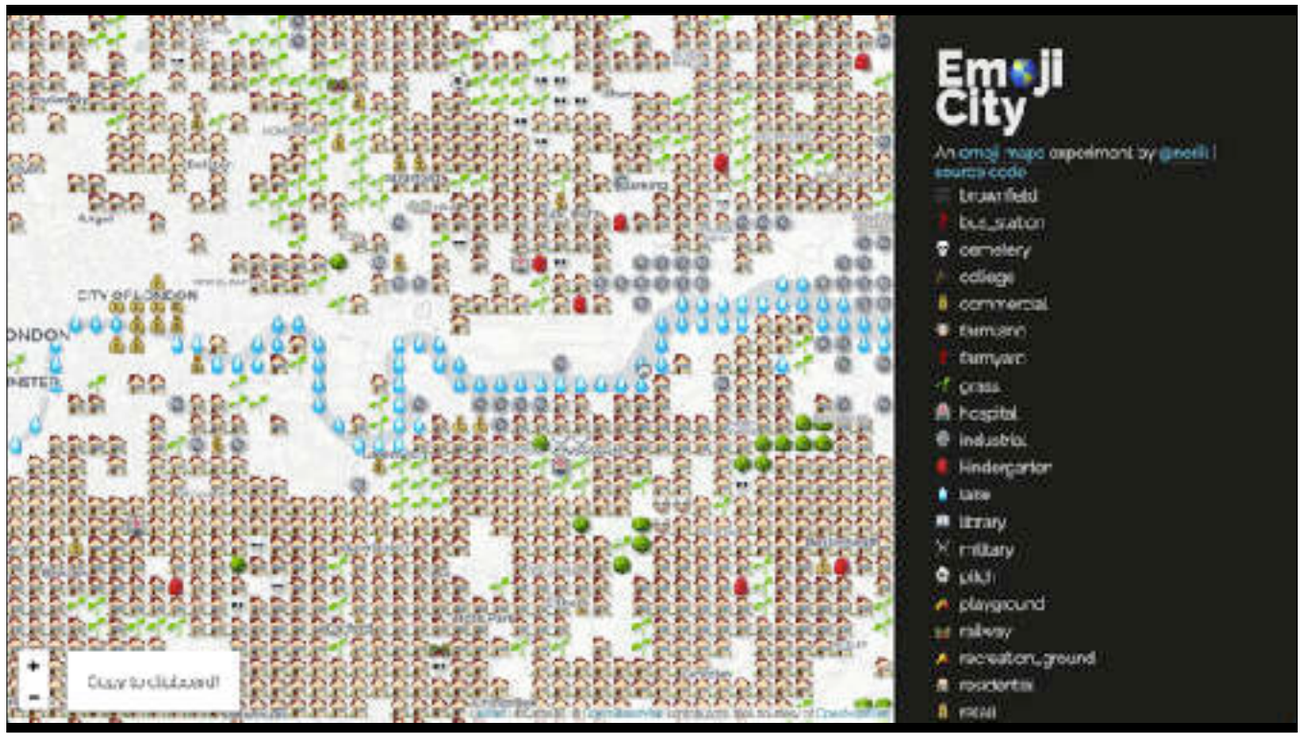


So we hook into Leaflet’s tile system, decode each tile with the vector-tile library, convert it into GeoJSON, then finally render it into a canvas Leaflet GeoJSON layer.

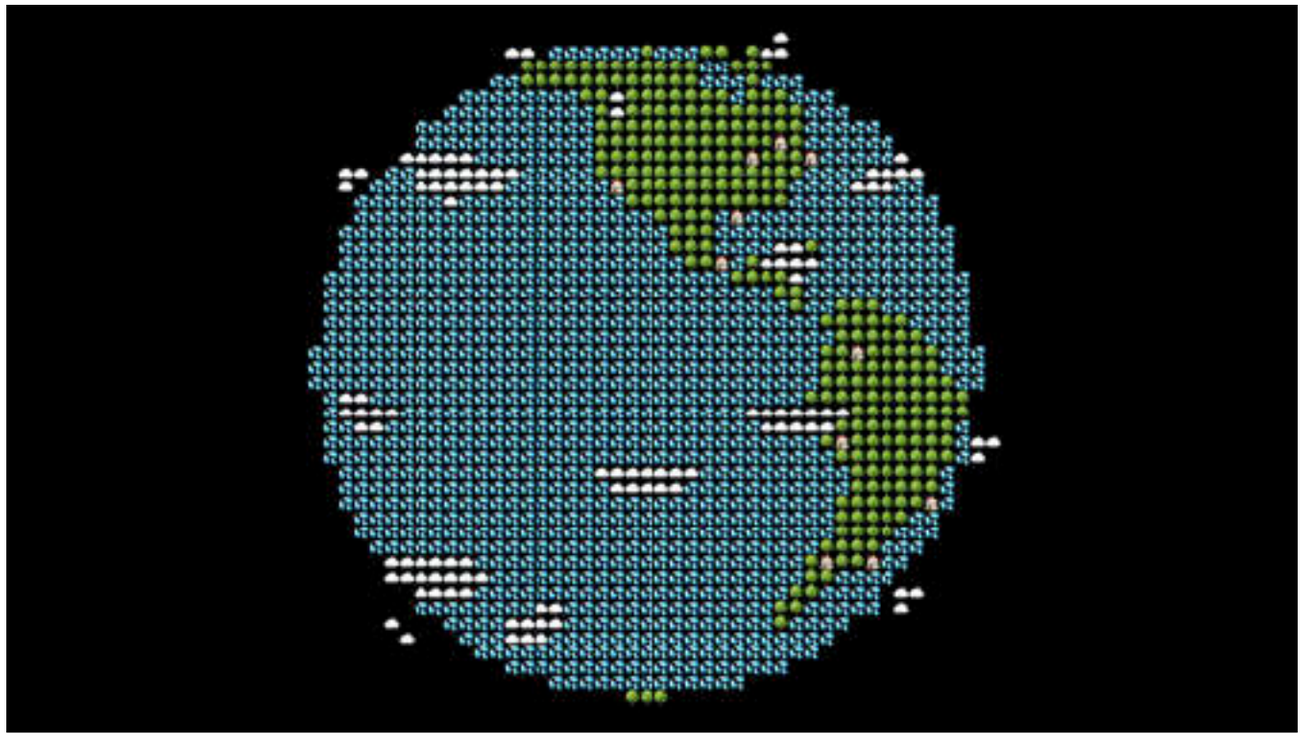
```
L.VectorEmoji = L.GridLayer.extend({
  //...
  createTile: function(coords, done) {
    //...
    var tileUrl = L.Util.template(this.__url, L.extend(data, this.options));
    var vectorTilePromise = fetch(tileUrl).then(function(response) {
        //...
        var pbf = new Pbf(reader.result);
        var vt = new VectorTile(pbf);
    }).then(function(json){
      for (var layerName in json.layers) {
        if (['landcover','landuse','water','natural'].indexOf(layerName) === -1) {
          delete json.layers[layerName];
          continue;
        }
        for (var i=0; i<json.layers[layerName].length; i++) {
          var feature = json.layers[layerName].feature(i);
          var geoJSONFeature = feature.toGeoJSON(coords.x, coords.y, coords.z);
        }
        // collect features in some global object...
      }
      // ...
    }.bind(this));
    //...
  }
});

var url = 'https://free-0.tilehosting.com/data/v3/{z}/{x}/{y}.pbf.pict?key=...';
var vectorGrid = new L.VectorGrid(url, vectorTileOptions).addTo(map);
```

This is roughly how it looks, a custom implementation of L.GridLayer (loosely based on the L.VectorGrid plugin)

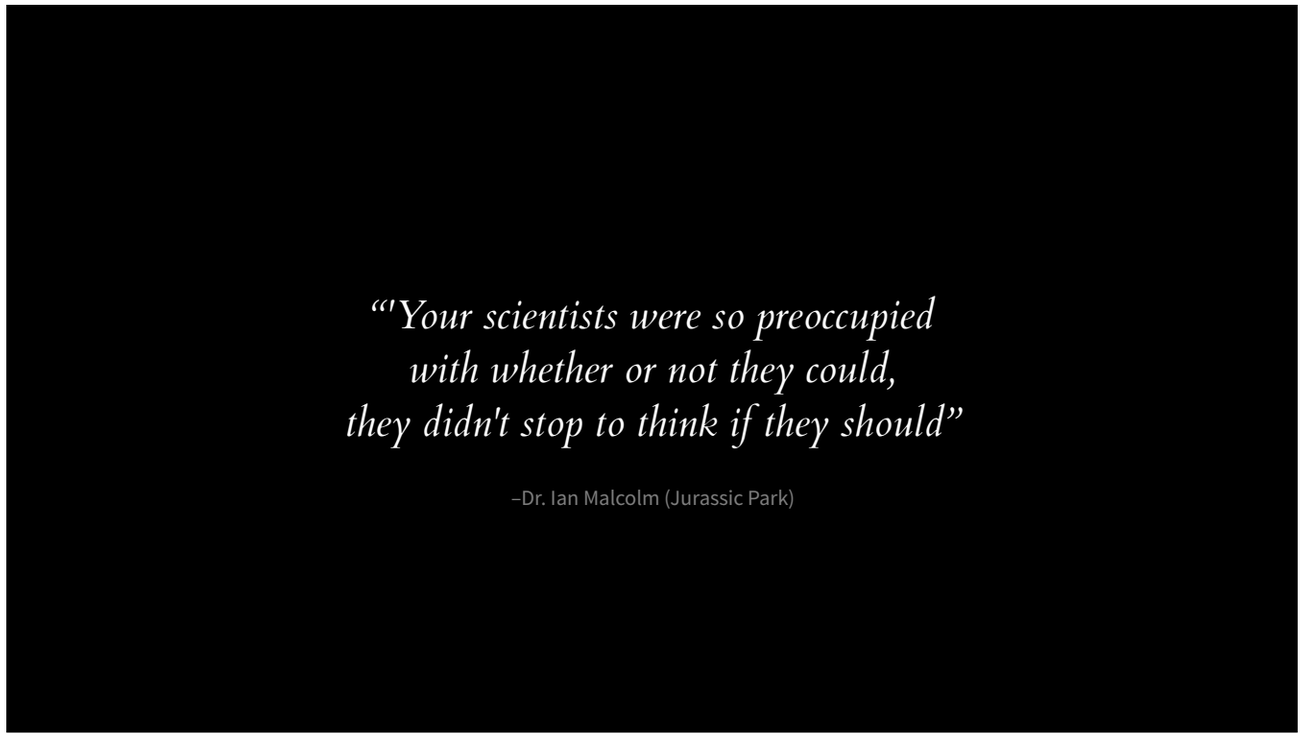
As I said we render those landuse/landcover polygons into a 2D canvas. Each value of the property we're interested in gets encoded into a random color. The final emoji grid is then rendered by picking a pixel color for each position of the grid.



This is a performance trick and allows for decent performance on a slippy map. Now we have a global, fractal, emoji map of the world. Go check out where you live, copy paste that map, make it yours.



One cool side effect with that canvas pixel picking technique, is that it works with anything that can be rendered into a canvas; here this is a 3D globe rendered with D3.



Yes that is bit pointless maybe, but that’s kind of the point of this.

You can learn a lot of things
with a stupid experiment

You just need a simple idea to unravel

Stupid makes us bold

I am just an engineer. Not a statistician, nor a geographer, I’m not even trained in GIS. Hiding behind the frivolous encouraged me to explore more, ask more questions, and dare.

We need lighter ways to talk
about hard problems

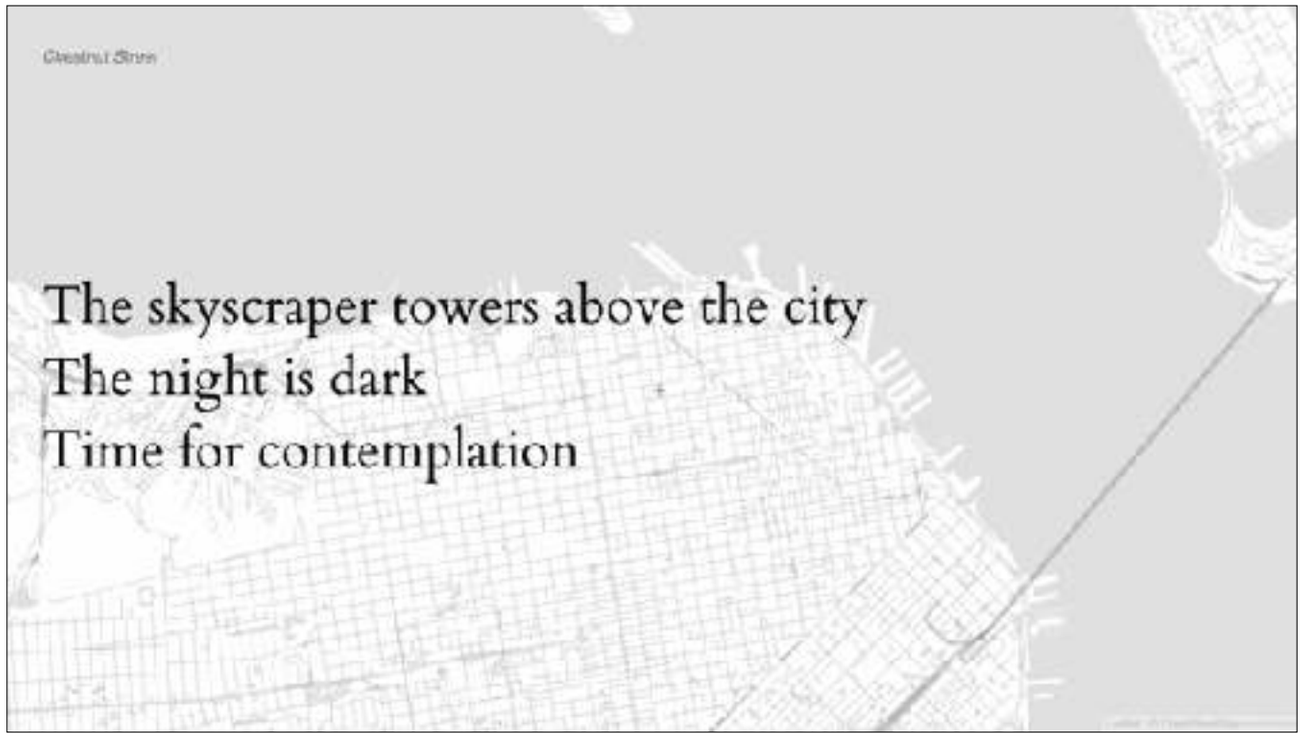
There’s no way a dashboard can’t communicate the urgency of climate change. We need to talk to more people.

Bonus Track:
OpenStreetMap Haiku

I’m done with emojis, and I have a few minutes to talk about another quirky side project we’re currently working on

There’s this amazing project called “Everything Every time”, form Naho Matsuda, a german-japanese artist. And the idea is to create automated poetry from the way we interact with our cities.

Technically, four locations in Manchester, equipped with a bunch of sensors, feeding an algorithm that generates semi random poetry.



We were captivated by this idea and the name “OpenStreetMap Haiku” started to feel like an evidently cool experiment to do. Instead of data streams from sensors, we’d use the gigantic amount of data in OSM to create automatic haikus for any place in the world.

We used the Overpass API that can pretty easily give you a list of “everything that surrounds you”

Generating a haiku

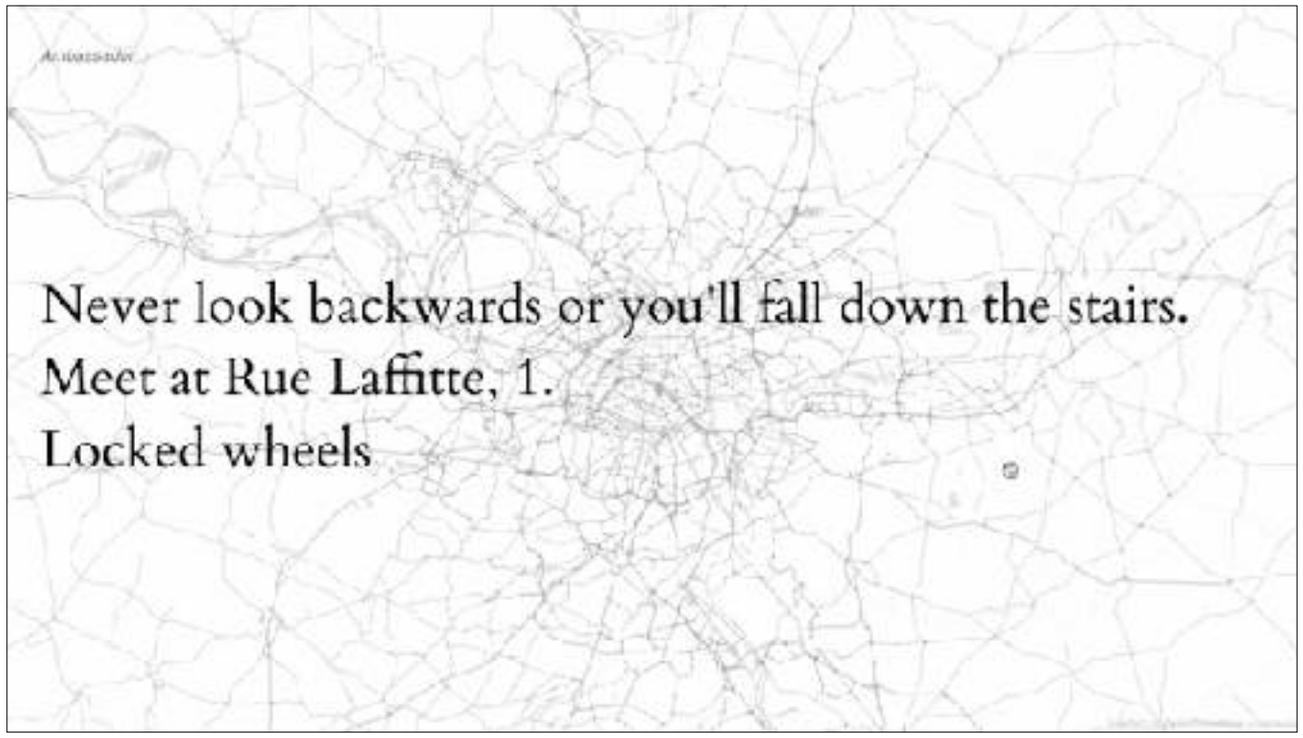
```
  {
    template: 'The boat arrives late again',
    tags: [['route','ferry']]
  },
  {
    template: ['A supermarket hustle and bustle', 'Salad cabbage and carrots’,
'The cashier\'s bored'],
    tags: [['shop', 'supermarket']]
  },
  {
    template: 'The skyscraper towers above the city’,
    tags: [['ele', '*']],
    condition: (el, env) => parseInt(el.tags.ele) >= 12
  }
```

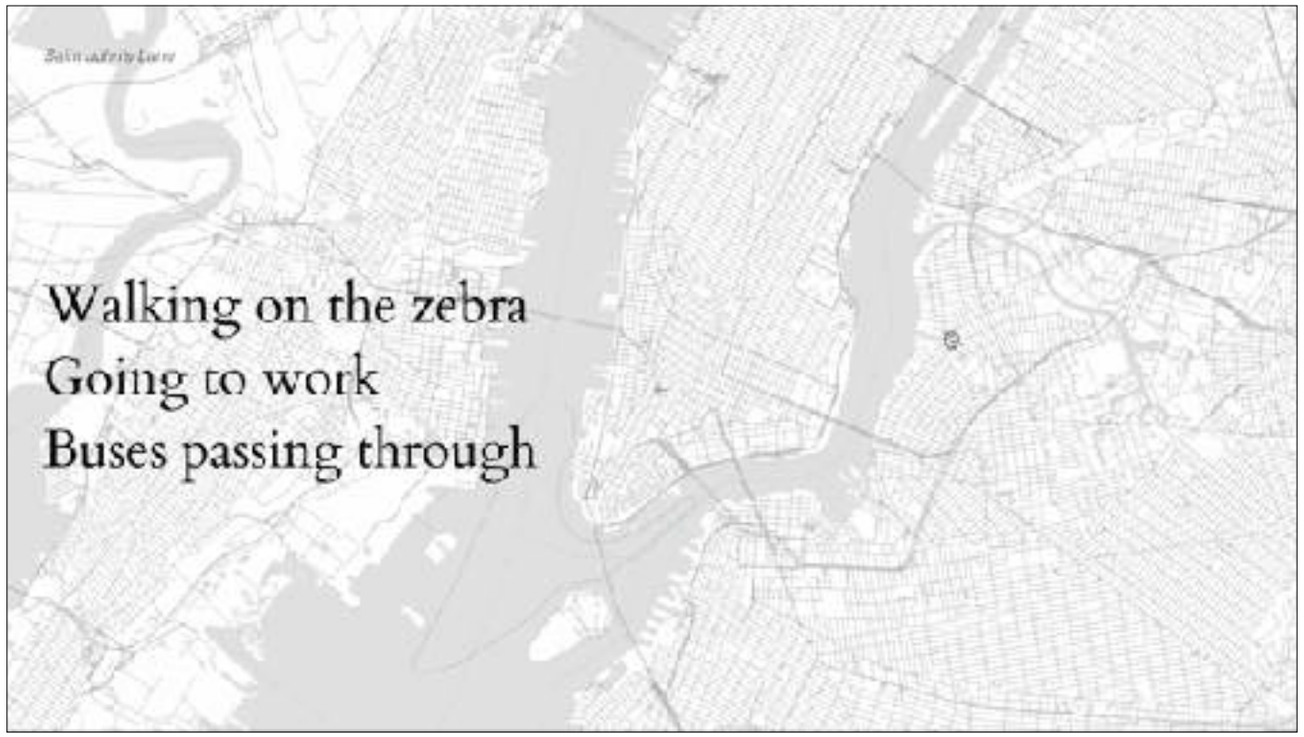
Then comes writing the poem per se. Nothing very fancy here, no AI: just a bunch of sentences, as much as we can write, with a few simple rules.

Generating a haiku

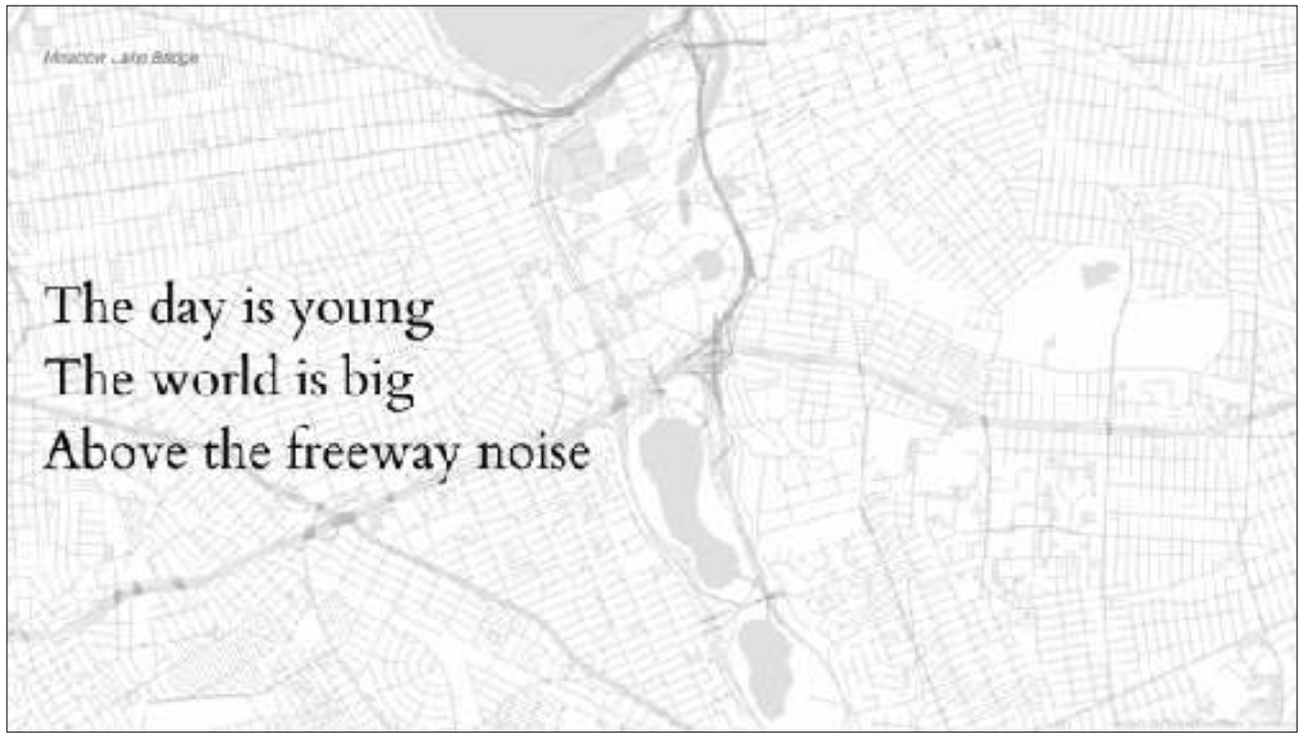
```
  {
    template: 'Heat on the pavement stones',
    tags: [['surface', 'paving_stones']],
    condition: (el, env) => env.temperature > 20
  },
  {
    template: (el, env) => `The smell of fresh coffee from ${el.name}`,
    tags: [['amenity', 'cafe']],
    condition: (el, env) => env.moment === 'morning',
    needsName: true
  },
  {
    template: ['Wet to the bone', 'Rain on the road like a mirror'],
    condition: (el, env) => env.weatherConditions.rain ||
env.weatherConditions.drizzle,
  }
```

Then comes writing the poem per se. Nothing very fancy here, no AI: just a bunch of sentences, as much as we can write, with a few simple rules.


Never look backwards or you'll fall down the stairs.
Meet at Rue Laffitte, 1.
Locked wheels



Adding sentences is really a fun game :)



I’ll just leave this here with some generative zen.

That’s all I got, thanks for listening. We are Satellite Studio, a young information design and creative mapping studio based in Madrid
http://satellitestud.io

Beyond experiments we build experiences, we are designers and coders, crafting beautiful and meaningful work. Please check out our work!
Slides and a repo with code are available at the URL at the bottom of the screen. Get in touch with me here or via twitter @nerik.