
Boundless
Vector Tiles with GeoServer
and OpenLayers

David Blasby
Canada

Engineering Lead
Boundless

Andreas Hocevar
Austria

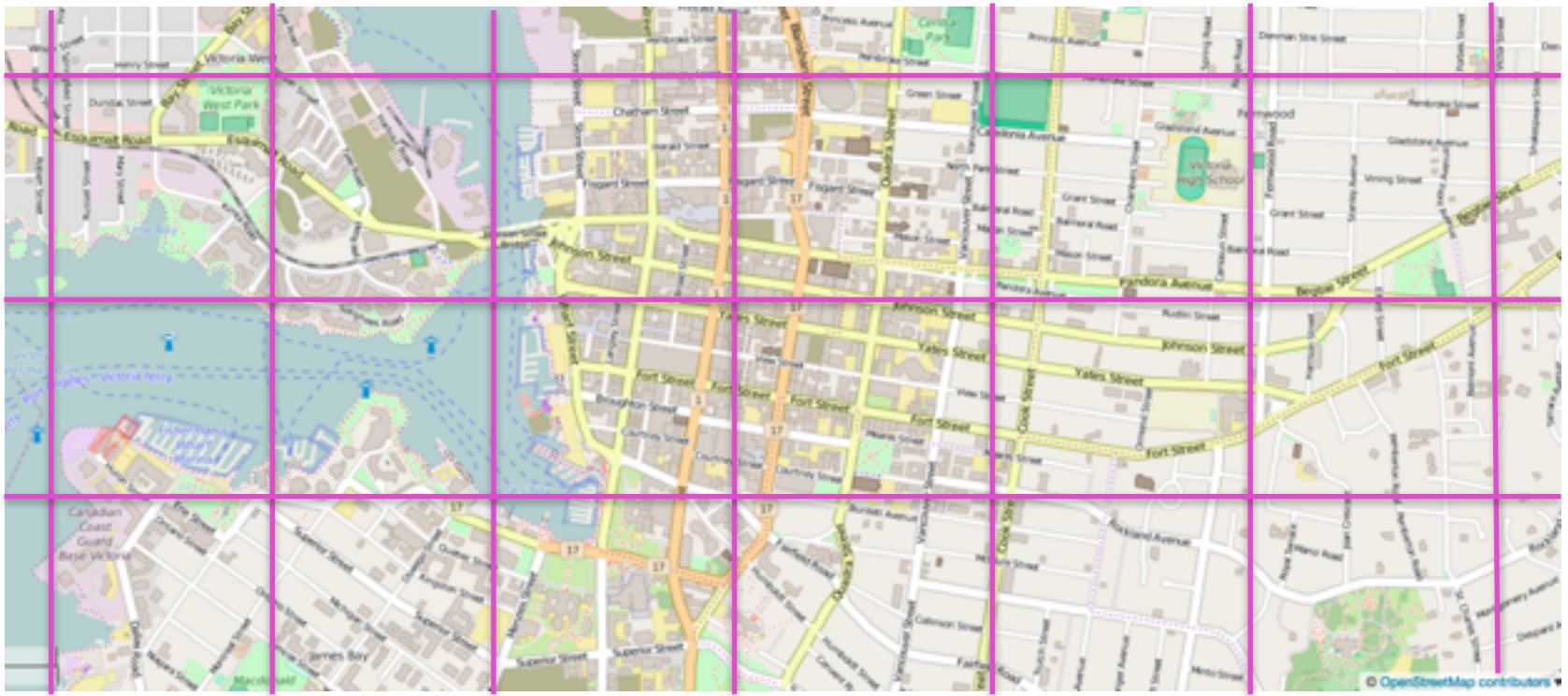
OpenLayers Engineer
Boundless

Gabriel Roldan
Argentina

Professional Services
Boundless

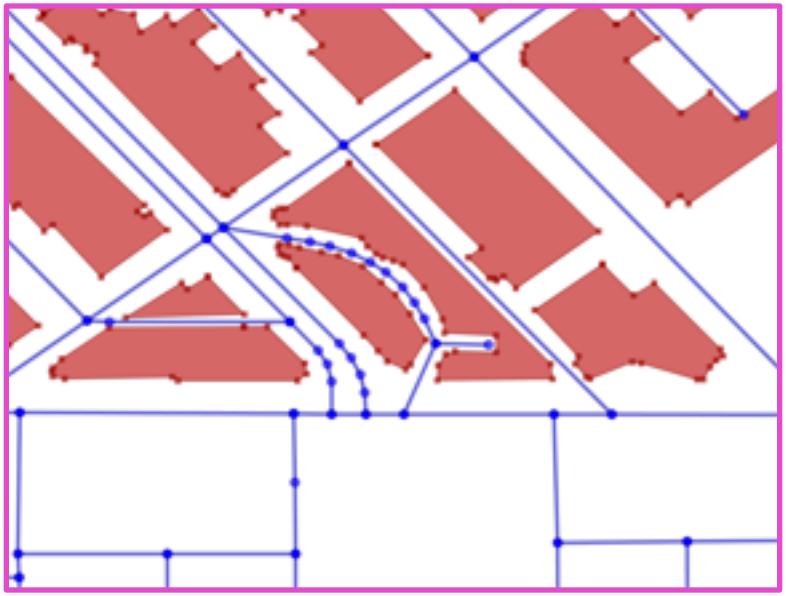
We're Hiring!

Image Tiled Map



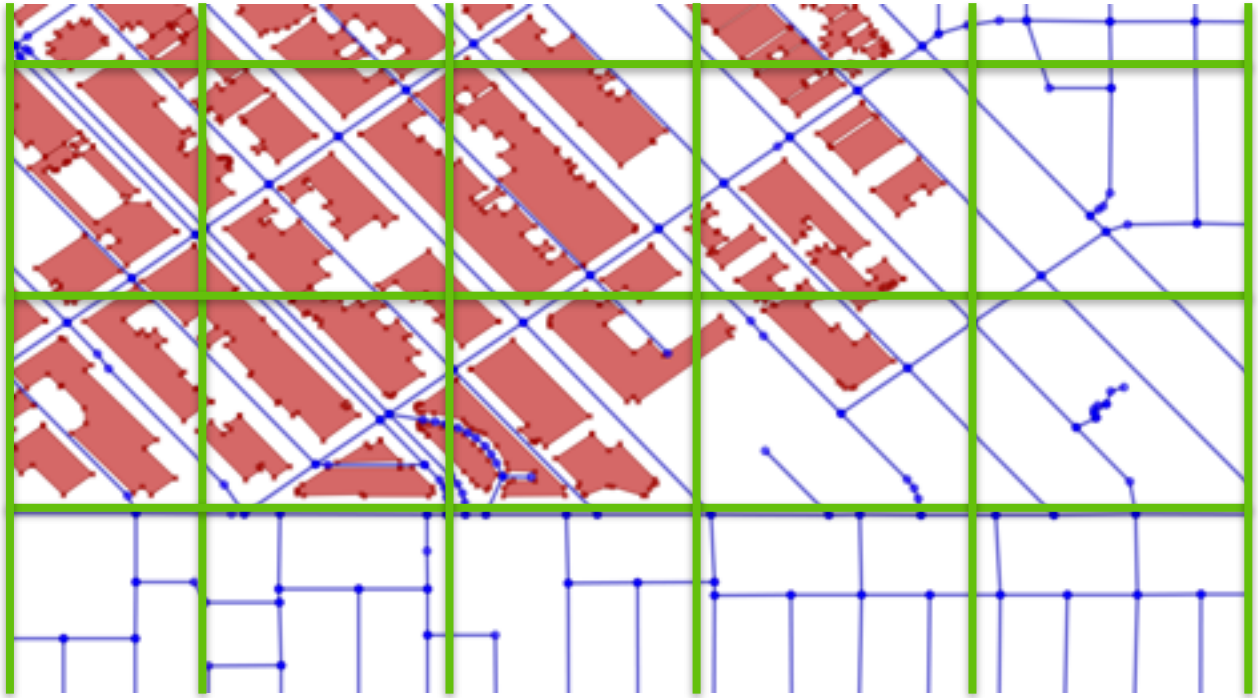
Single Open Street Map Image Tile

A Tile of feature data



Real geometry and attribute data
(GeoJSON, TopoJSON, MVP, …)

Vector Tiles



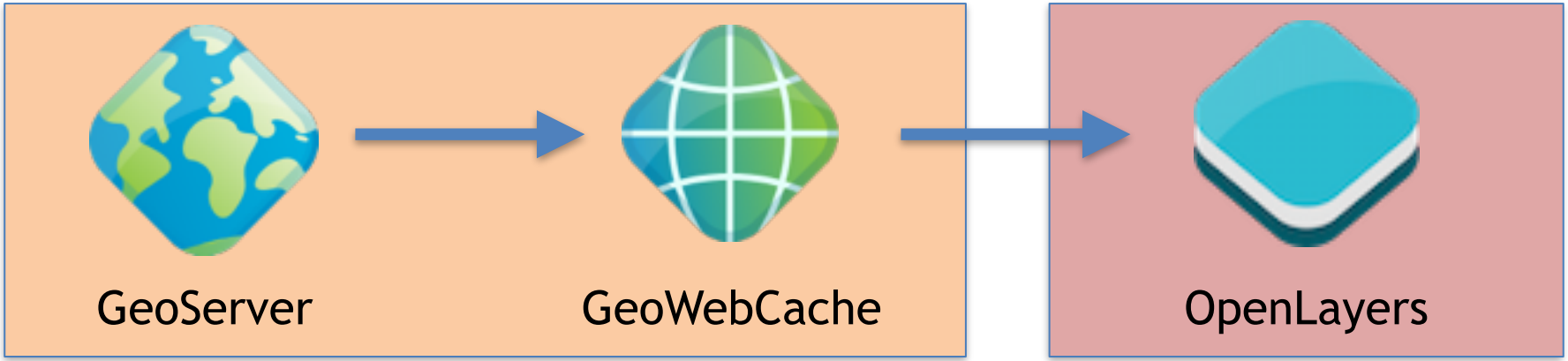
Vector Tiles versus Image Tiles

1. Client (not the server) decides on styling
2. Only need to tile the data once to have multiple maps
3. Drawn vectors can look better on high-resolution displays
4. Image Tiles are much easier to consume
5. There's more know-how in working with Vectors

Empowering

Efficient

Outline of this talk

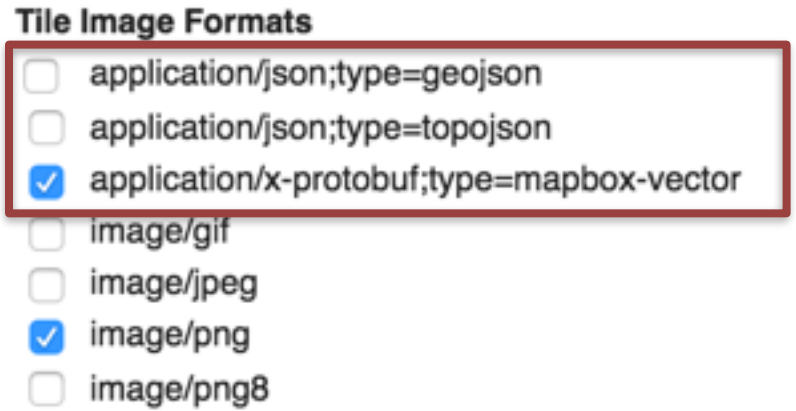


- 5 seconds to turn on Vector Tiles
- Vector Tiles in the OGC context
- Geoserver Rendering Process
- SLD to control Vector Tile Generation

- OpenLayers Overview
- Vector Tile Maps
- Styling
- Advanced user interaction
- Demo

5 Second to turn on Vector Tiles in Geoserver

```
…&SERVICE=WMS&REQUEST=GetMap&FORMAT=application/x-protobuf;type=mapbox-vector
```



Vector Tiles in the OGC Services Context

- **WMS** - Creating Maps
- **WMTS** - Tiling
- **WFS** - Feature Access

WFS and Vector Tiles

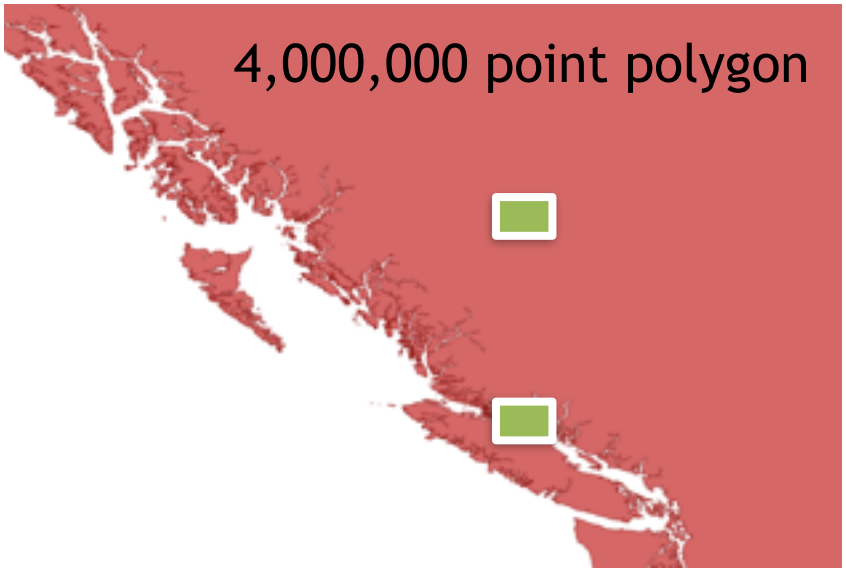
- Both return unstyled vector/attribute data

- WFS returns the underlying data *unmodified*
- Vector Tiles return *modified* (ready-to-render) features

WFS: Glorious Detail

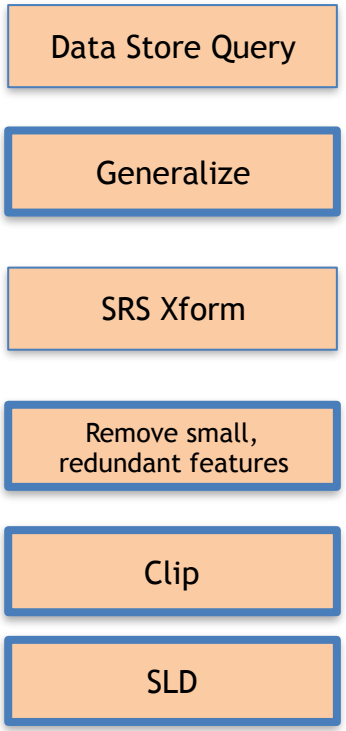
VTs: Easy to render

Couldn’t I just use the WFS to generate VTs?



It’s just not practical!

GeoServer Rendering Process Overview



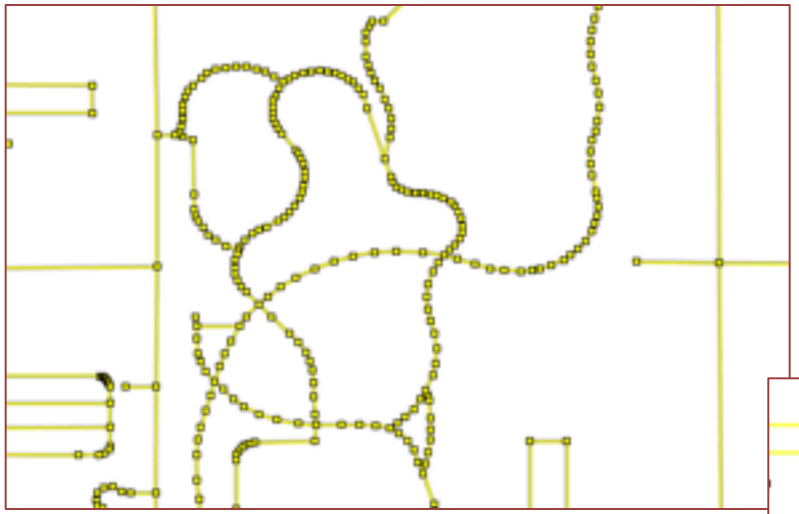
GeoServer's WMS has several different renders, including;

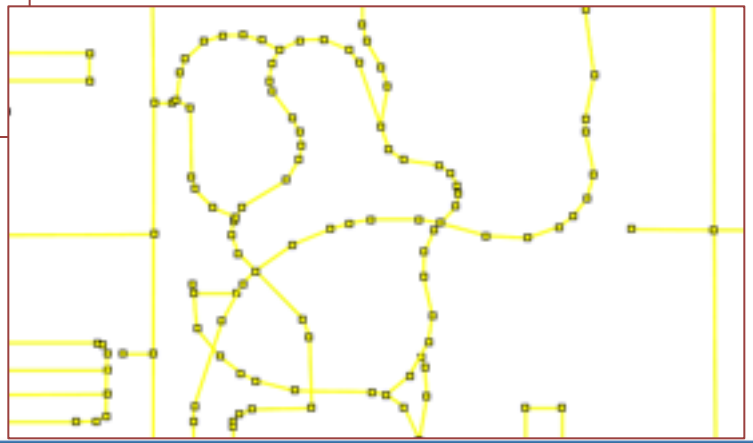
a) Streaming Renderer - used to make image maps
b) Vector Tiles Renderer - used to make vector tiles

They work almost the same!

Generalization/Simplification

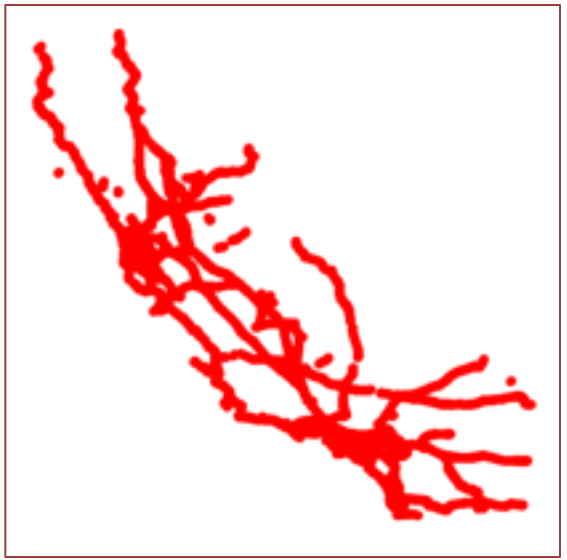
- Data Store Query
- **Generalize**
- SRS Xform
- **Remove small, redundant features**
- **Clip**
- SLD





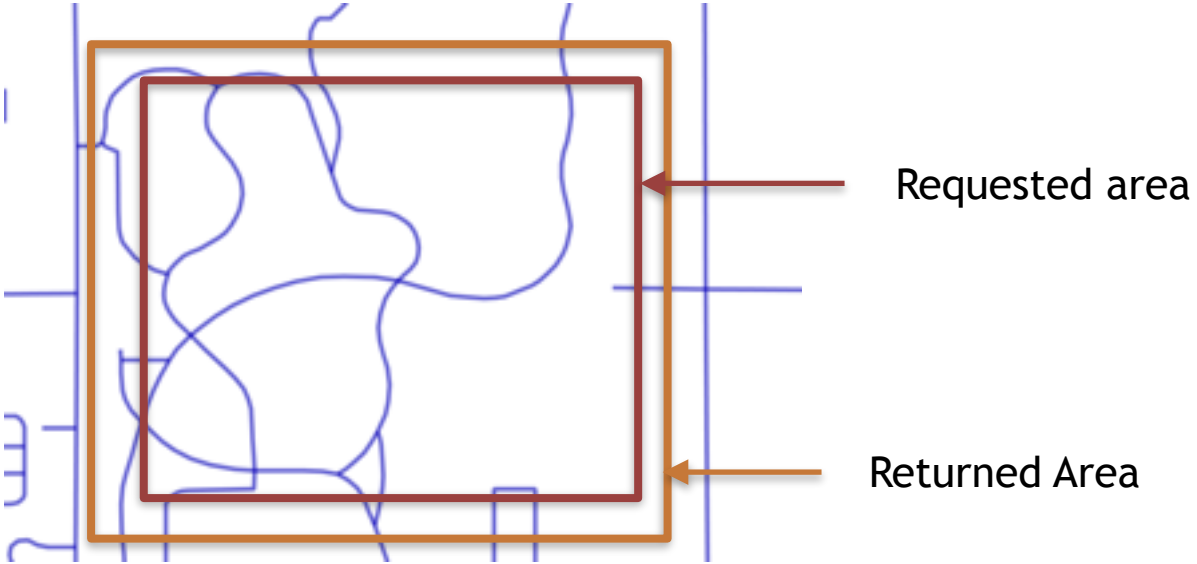
Removing small, redundant features

- Data Store Query
- **Generalize**
- SRS Xform
- **Remove small, redundant features**
- **Clip**
- SLD

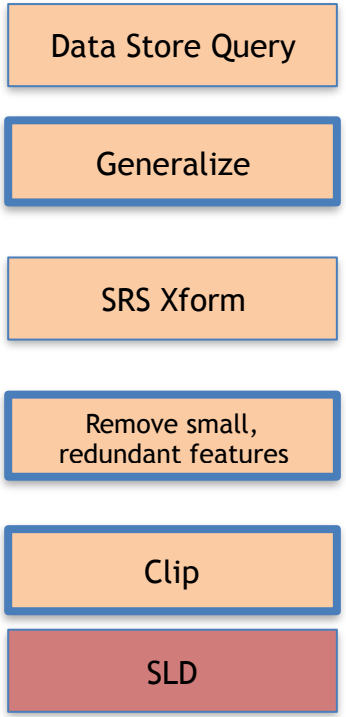


Clipping

- Data Store Query
- Generalize
- SRS Xform
- Remove small, redundant features
- Clip
- SLD



SLD: Controlling what's drawn



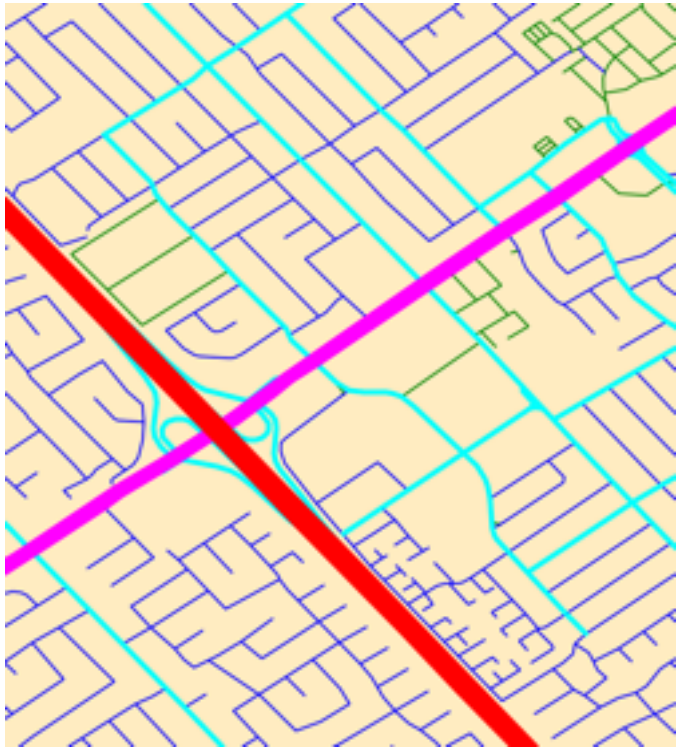
In GeoServer, use SLD to control map styling.

Three most important parts of SLD styling rules:

1. Scale
2. Filter
3. Actual style information (stroke/fill)

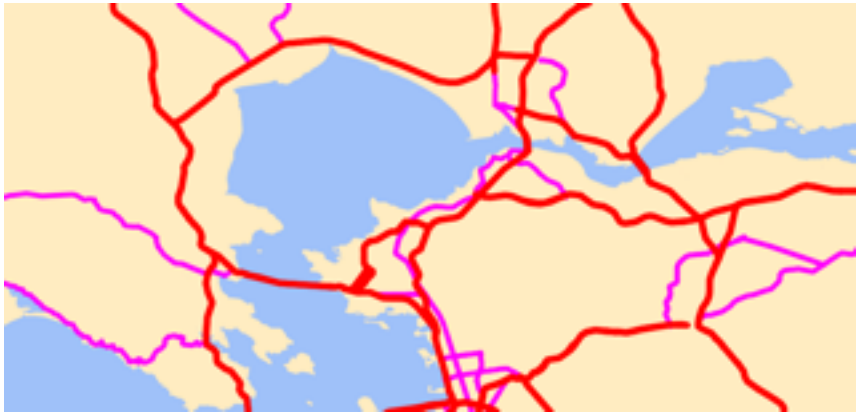
Controls what's queried and "rendered"!

The world's ugliest map



```
- name: residential roads
  rules:
  - filter:  type='residential'
    name: residential roads
    scale: [0, 70000]

    symbolizers:
    - line:
        stroke-color: blue
        stroke-width: 1
        stroke-linecap: round
```

When the map scale is more than 1:70,000 - include the *residential* roads.



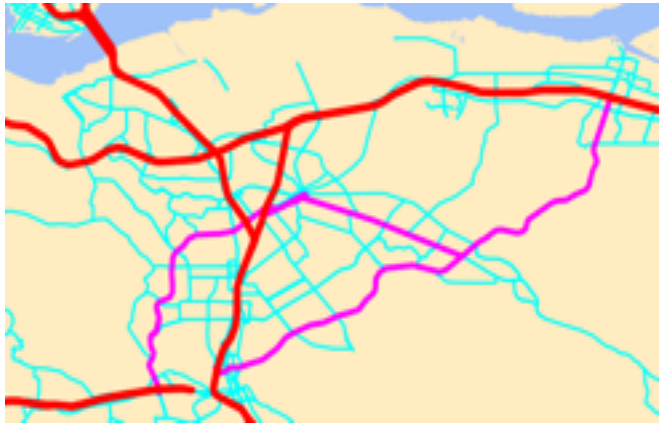
Tile Image Formats

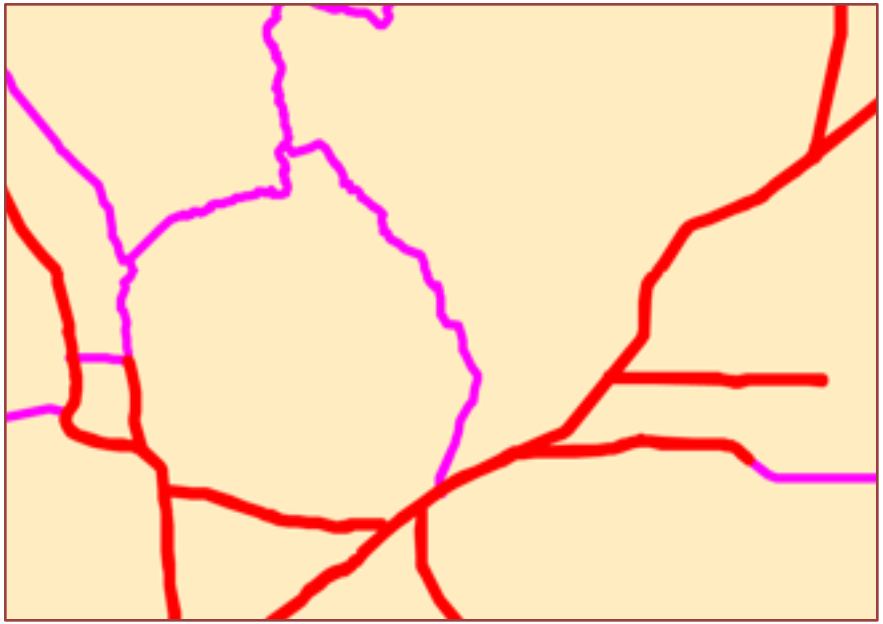
- [ ] application/json;type=geojson
- [ ] application/json;type=topojson
- [x] application/x-protobuf;type=mapbox-vector
- [ ] image/gif
- [ ] image/jpeg
- [x] image/png
- [ ] image/png8

Tick!

Image Tiled Map

Vector Tiles Map (same styling)





Changing the OpenLayers Style

Wait! Where's the Footways?

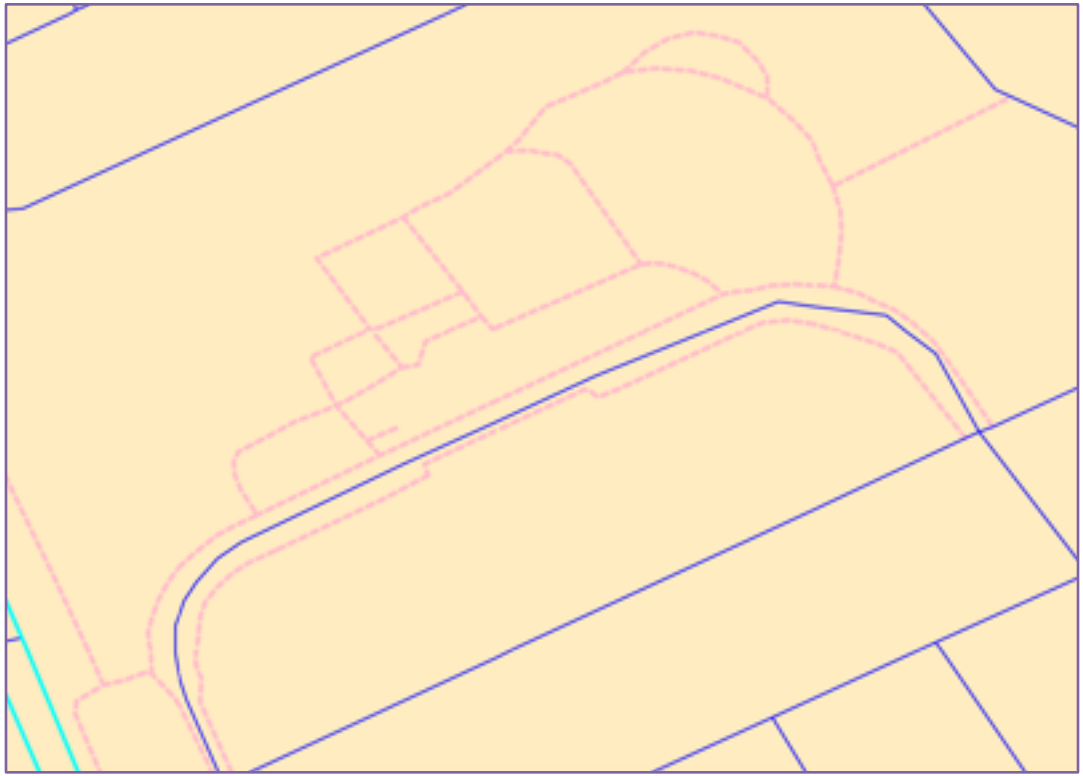
```
name: OSM Roads
title: OSM Roads
abstract: OSM Roads
feature-styles:

- name: samll roads
  rules:
  - filter:  type='service' or  type='path' or  type='track' or  type='unclassified' or
type='living_street' or  type='road'
    name: samll roads
    scale: [0, 40000]
    symbolizers:
    - line:
        stroke-color: green
        stroke-width: 1
        stroke-linecap: round


- name: medium roads
  rules:
  - filter:  type='residential'
    name:  medium roads
    scale: [0, 70000]
    symbolizers:
    - line:
        stroke-color: blue
        stroke-width: 1
        stroke-linecap: round


- name: secondary roads
  rules:
  - filter:  type='secondary' or type='tertiary' or type='motorway_link' or type='primary_link'
type='secondary_link' or type='tertiary_link'or type='motorway_link'
    name: secondary roads
    scale: [0, 200000]
    symbolizers:
    - line:
        stroke-color: cyan
        stroke-width: 3
        stroke-linecap: round
```

No rule for `type='footway'`

```
- name: footways
  rules:
  - filter:  type='footway'
    name: footways
    scale: [0, 40000]
    symbolizers:
    - line:
        stroke-color: pink
        stroke-width: 4
        stroke-linecap: round
```

GeoWebCache will automatically regenerate the cache.

FootWays shown with dashed style (OpenLayers)



OpenLayers

OpenLayers: Map everything

- Images, image tiles, vector data, tiled vector data
- Any projection
- Any orientation -> full rotation support
- Animations
- Integrate with e.g. Cesium or d3

If it has location, OpenLayers can render it!

Vector tiles in OpenLayers

- Mapbox vector tiles preferred (optimized for rendering)
- All vector formats supported
- Same styling as untiled vector data
- Interactive maps - access to feature attributes

Not to be used as replacement for vector (as in WFS) data!

Mapbox Vector Tiles Support

- `ol.format.MVT`
- Uses Mapbox's pbf library to read the binary tile data
- Uses Mapbox's vector-tile library to extract layers and features
- Configurable to only read a subset of the available layers
- Creates lightweight `ol.RenderFeature` or standard `ol.Feature` features with pixel coordinates

How to create a vector tile layer

// The OGC way, step 1: WMTS from capabilities

```
var caps = new ol.format.WMTSCapabilities().read(data);
var wmts = new ol.source.WMTS(
    ol.source.WMTS.optionsFromCapabilities(caps, {
      layer: 'opengeo:california',
      matrixSet: 'EPSG:3857',
      format: 'application/x-protobuf;type=mapbox-vector'
    })
);
```

How to create a vector tile layer

```
// The OGC way, step 2: url and tilegrid from WMTS

var layer = new ol.layer.VectorTile({
  source: new ol.source.VectorTile({
    format: new ol.format.MVT(),
    tileUrlFunction: wmts.getTileUrlFunction(),
    tileGrid: wmts.getTileGrid()
  }),
  style: function(feature, resolution) { /* ... */ }
});
```

Style streets nicely

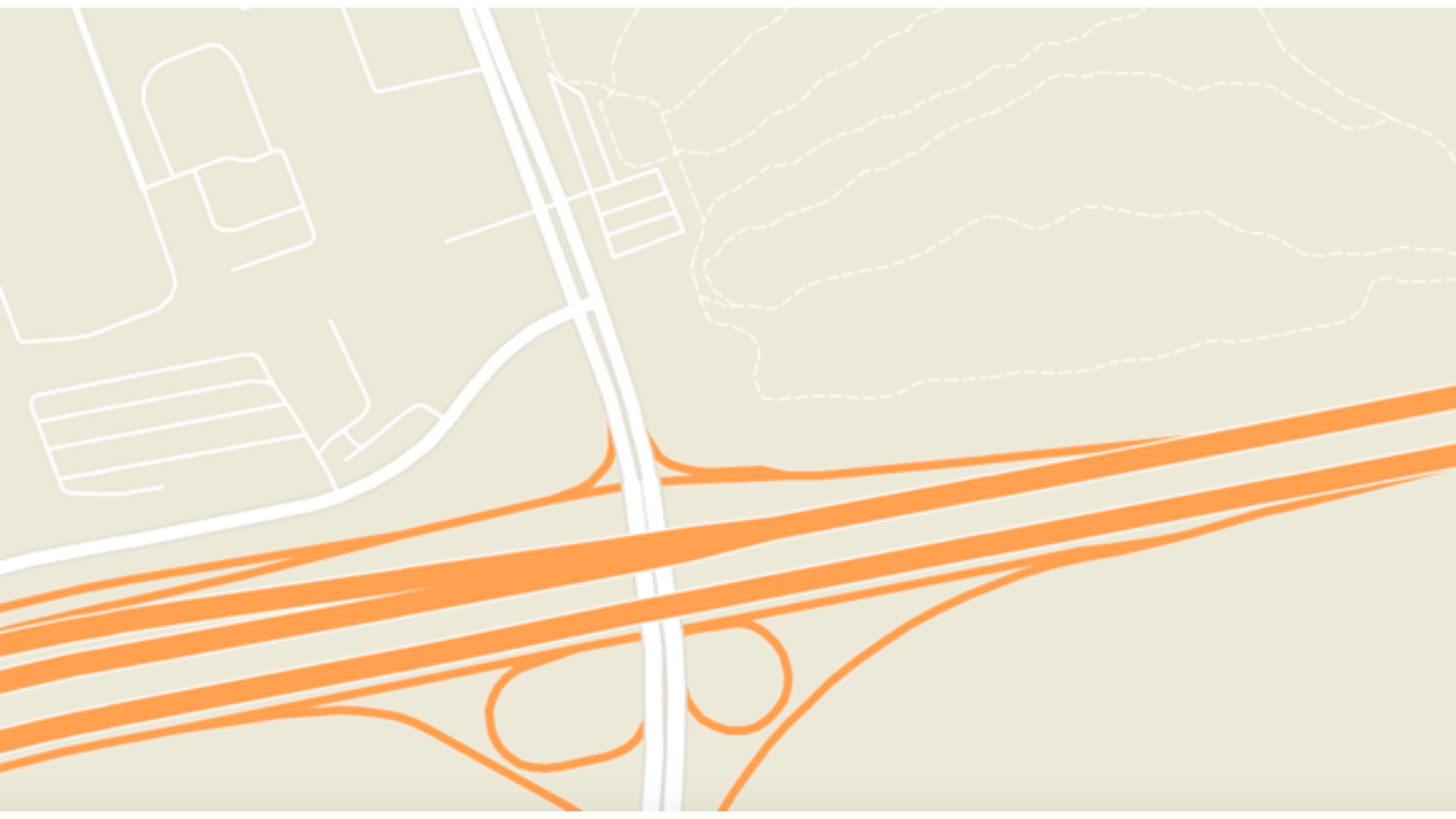
```
[
  new ol.style.Style({
    zIndex: 1,
    stroke: new ol.style.Stroke({color: '#fff', width: 4})
  }),
  new ol.style.Style({
    zIndex: 2,
    stroke: new ol.style.Stroke({color: '#ddd', width: 3})
  })
]
```

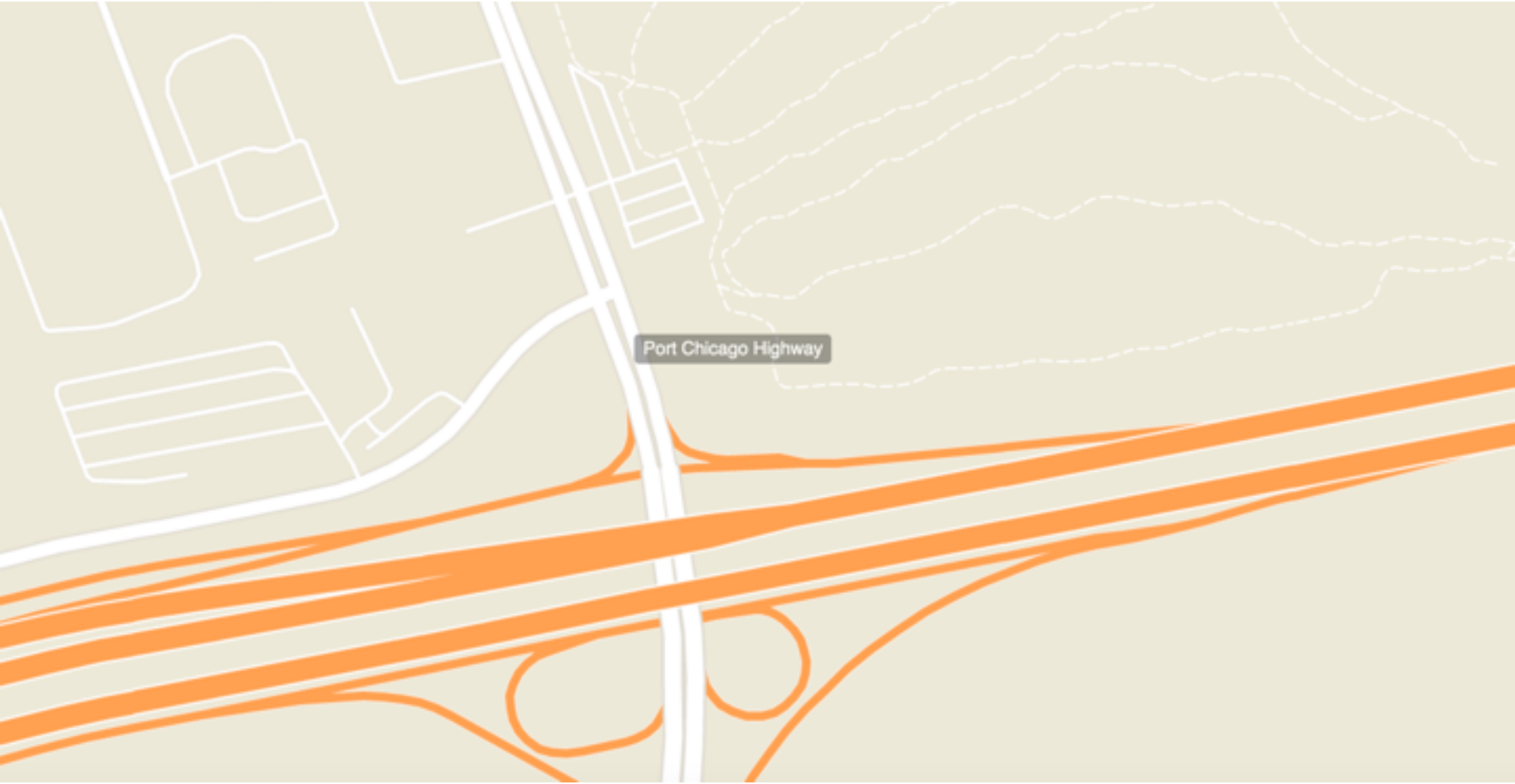
Interactivity - info on hover

```
var info = document.createElement('div');
var overlay = new ol.Overlay({element: info});
map.addOverlay(overlay);
map.on('pointermove', function(e) {
  var name = map.forEachFeatureAtPixel(e.pixel, function(feature) {
    return feature.get('name');
  });
  info.style.display = name ? '' : 'none';
  info.innerHTML = name;
  overlay.setPosition(e.coordinate);
});
```

OpenLayers Demo

Q & A




Port Chicago Highway