

Web Mapping and Geovisualisation

Data Architecture and Formats

Matthew Howard, Elisabetta Pietrostefani and Dani Arribas-Bel

Today

- Spatial Data
- Spatial Data Formats (old and new)
- Selecting the right format

Spatial Data

The data we use to represent space falls into one of two categories:

- **Vector**
- **Raster**

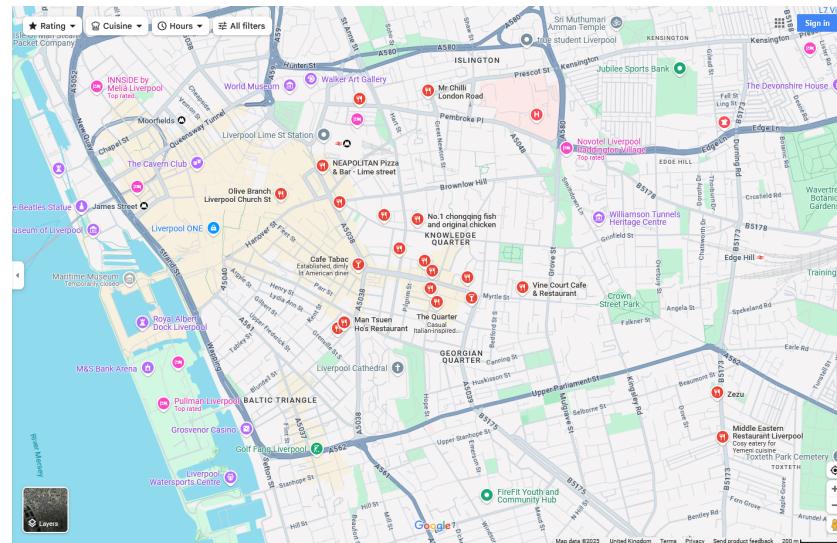
Vector

Space is divided into a finite set of entities, each of which is represented by a shape or geometry...

- (Multi-)Points
- (Multi-)Lines
- (Multi-)Polygons

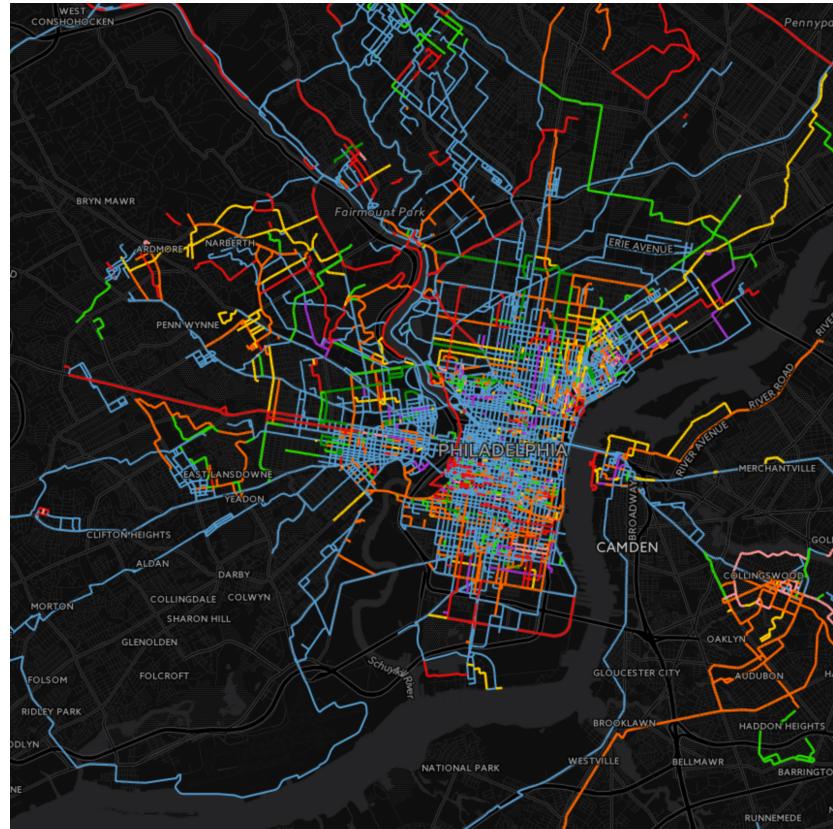
A few examples

Points



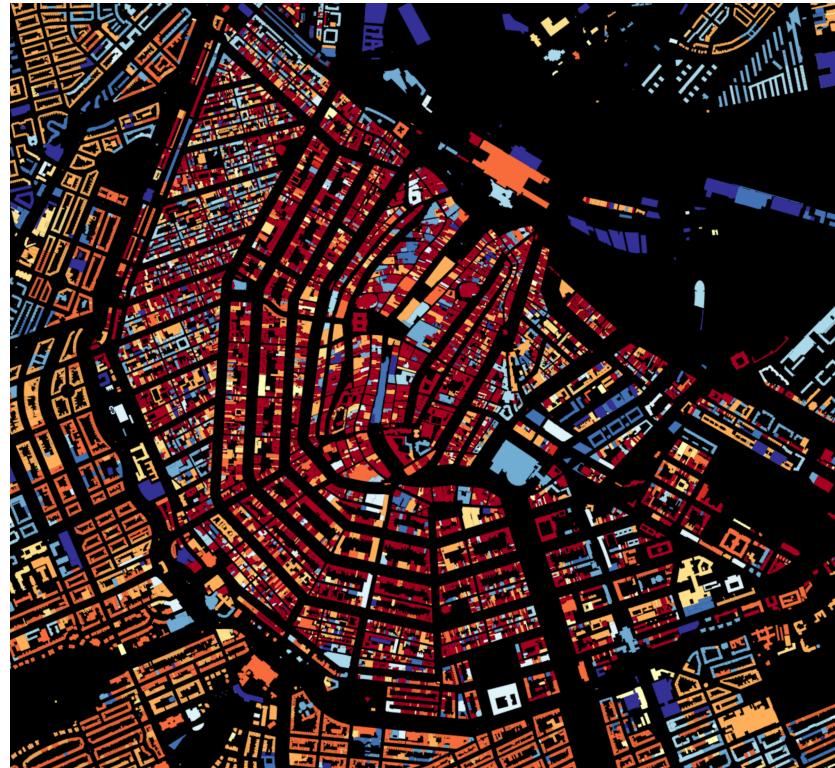
Source

Lines



Source

Polygons



Source

Combinations



Source

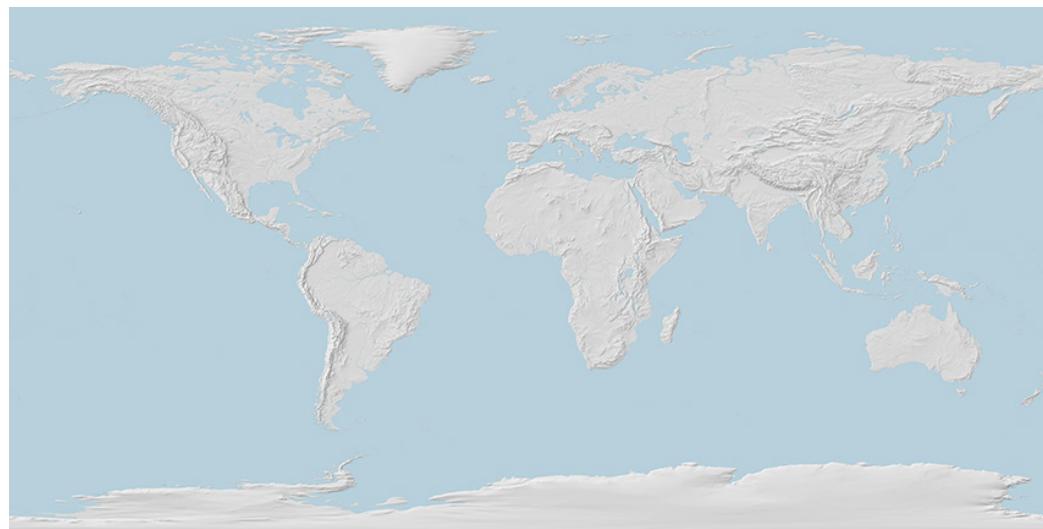
Raster

Use an image and control pixel colors to encode value

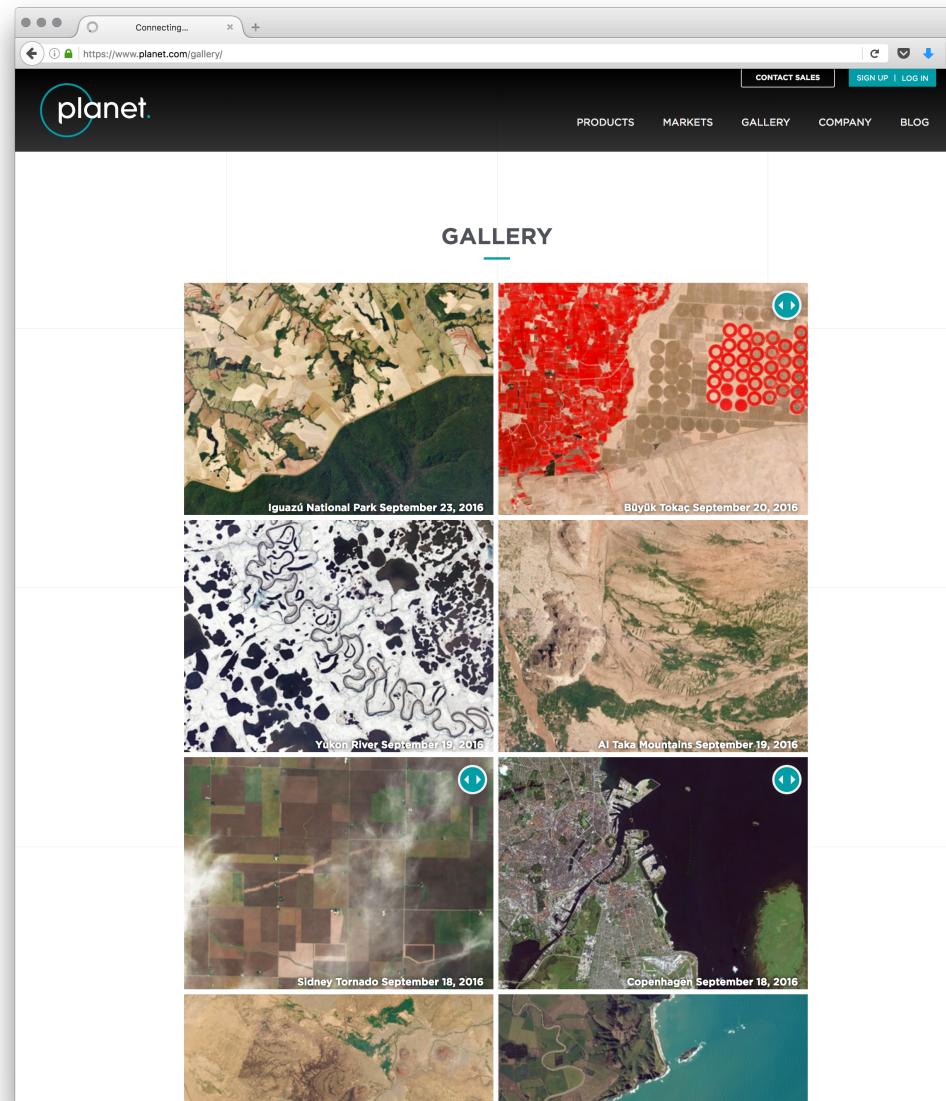
The value assigned for each cell represents the attribute of that cell, i.e:

- Continuous variables (temperature, elevation etc.)
- Satellite Imagery (e.g. land use/cover)

Continuous Example



Satellite Example



Spatial Data Formats

In general...

- Points, lines, polygons -> Vector Formats
- Images, surfaces -> Raster Formats

Traditional formats

Vector - Individual entity type files: [shapefiles](#) etc.

Raster - Mostly (single) image formats: [GeoTIFFs](#) etc.

However...

many of these formats were designed for an *offline* world, so display some (or all) of the following:

- Binary
- “Unqueriable”
- complex structures

The screenshot shows a web browser window with the title "Shapefile must die!" and the URL "switchfromshapefile.org". The main content is titled "Switch from Shapefile". It discusses the history and limitations of ESRI Shapefiles, mentioning they have been around since the early 1990s and are the most commonly used vector data exchange format. It highlights that while Shapefiles have enabled many successful activities, they also have limitations that complicate software development and reduce efficiency. Members of the geospatial IT industry believe it's time to stop using Shapefiles as the primary vector data exchange format and replace them with a format that takes advantage of modern advances.

Read more:

- The good side
- Shapefile is a bad format
- Shapefile alternatives

The good side

Shapefile does a lot of things right. Here are some reasons why Shapefile is so heavily used:

- Shapefile is by far the most widely supported format in existing software packages.
- While the format is proprietary, the [specification is open](#).
- For many use cases, it is *good enough*:
 - Index files (e.g. .shx) enable good reading performance.
 - It is relatively efficient in terms of file size. The resulting file, even un-zipped, is relatively small compared to some other (mostly text-based) formats.

Shapefile is a bad format

Why is Shapefile so bad? Here are several reasons why the Shapefile is a bad format and you should avoid its usage:

- No coordinate reference system definition.
- It's a multifeile format.
- Attribute names are limited to 10 characters.
- Only 255 attributes. The DBF file does not allow you to store more than 255 attribute fields.
- Limited data types. Data types are limited to float, integer, date and text with a maximum 254 characters.
- Unknown character set. There is no way to specify the character set used in the database.
- It's limited to 2GB of file size. Although some tools are able to surpass this limit, they can never exceed 4GB of data.
- No topology in the data. There is no way to describe topological relations in the format.
- Single geometry type per file. There is no way to save mixed geometry features.
- More complicated data structures are impossible to save. It's a "flat table" format.
- There is no way to store 3D data with textures or appearances such as material definitions. There is also no way to store solids or parametric objects.
- Projections definition. They are incompatible or missing.
- Line and polygon geometry type, single or multipart, cannot be reliably determined at the layer level, it must be determined at the individual feature level.
- Add more ...

Switch from shapefile

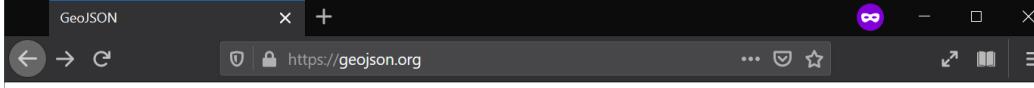
Modern Formats

New formats have appeared aiming to “fix” those issues, but also responding to web needs:

- Streamable(e.g. [GeoJSON](#))
- Querable (e.g. [PostGIS/Geopackage](#))
- Single files (e.g. [.mbtiles](#))

We’ll be focusing on 2 specific examples...

GeoJSON



The screenshot shows a web browser window with the title bar "GeoJSON". The address bar displays the URL "https://geojson.org". The main content area features a large blue header "GEOJSON". Below it is a sub-header: "GeoJSON is a format for encoding a variety of geographic data structures." A code block shows a JSON object representing a Feature:

```
{  
  "type": "Feature",  
  "geometry": {  
    "type": "Point",  
    "coordinates": [125.6, 10.1]  
  },  
  "properties": {  
    "name": "Dinagat Islands"  
  }  
}
```

Text below the code block explains: "GeoJSON supports the following geometry types: Point, LineString, Polygon, MultiPoint, MultiLineString, and MultiPolygon. Geometric objects with additional properties are Feature objects. Sets of features are contained by FeatureCollection objects."

A green button labeled "The GeoJSON Specification (RFC 7946)" is visible.

In 2015, the Internet Engineering Task Force (IETF), in conjunction with the original specification authors, formed a [GeoJSON WG](#) to standardize GeoJSON. [RFC 7946](#) was published in August 2016 and is the new standard specification of the GeoJSON format, replacing the 2008 GeoJSON specification.

Source

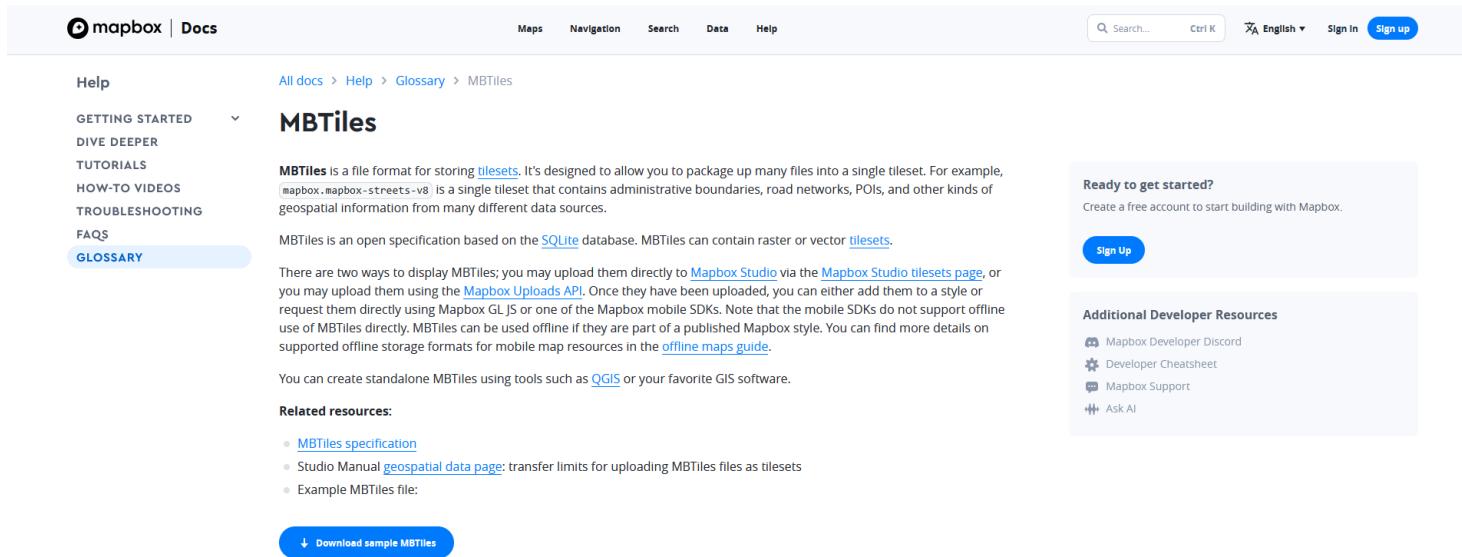
Advantages

- Plain text (Human readable)
- Streamable
- Well intergrated with web standards ([JSON](#))

Disadvantages

- Plain text (inefficient)
- Non-queriable
- Vector only

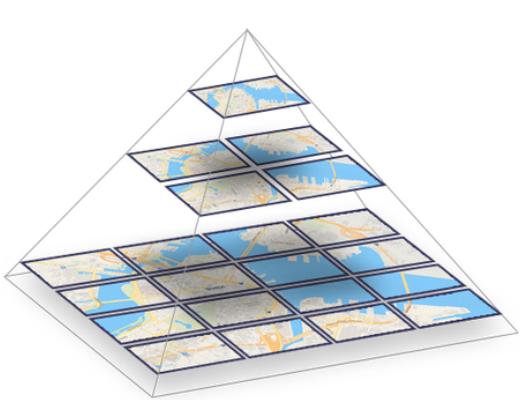
Tilesets (.mbtiles)



The screenshot shows a detailed view of the Mapbox Docs website. At the top, there's a navigation bar with links for Maps, Navigation, Search, Data, Help, a search input field, and language selection (English). Below the navigation is a main content area with a sidebar on the left containing links for Help, Getting Started, Dive Deeper, Tutorials, How-to Videos, Troubleshooting, FAQs, and Glossary (which is highlighted). The main content page is titled "MBTiles" and includes a breadcrumb trail: All docs > Help > Glossary > MBTiles. The page content explains what MBTiles is, how it's used, and provides links to related resources like the specification and studio manual. On the right side, there's a "Ready to get started?" section with a "Sign Up" button, and another section for "Additional Developer Resources" with links to developer discord, cheatsheet, support, and AI.

Source

Why do we use tiles?

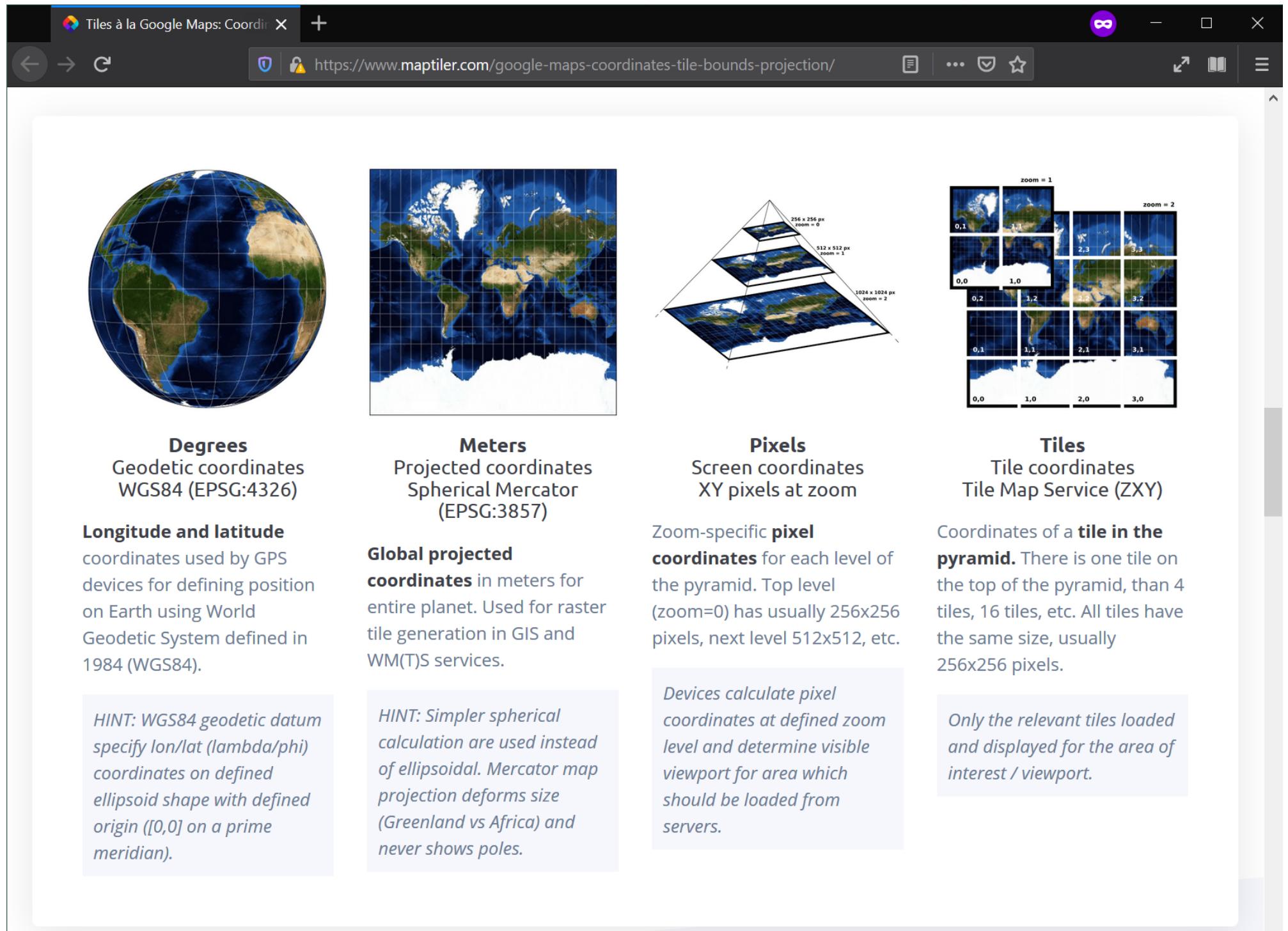


How does a zoomable map work?

People are using coordinate systems and map projections to transform the shape of Earth into usable flat maps for centuries.

A map of the entire world is too big to be directly displayed in a computer so there is a clever mechanism for quick browsing and zooming on maps: the map tiles.

The world is divided into small squares, each with fixed geographic area and scale. This clever trick allows you to browse just a small part of the planet without loading the whole map - and you still get an illusion of exploring a single huge document.



Advantages

- Querable (SQLite)
- Fast access to large maps with limited resources (client/server model + queriable format)
- Some (vector tiles) are stylable

Disadvantages

- Designed for webmap services *not* analysis (e.g. rasters are just images)
- A dataset needs to be stored at several zoom levels
- Once created, hard to modify (e.g. reproject)

Selecting a format

No silver bullet...

- What type of data do you want to store? Vector or raster
- What are you going to do with the file? Analysis or serving
- What environment are you working? Locally or web

Examples

- Analysing a large geotagged dataset of social media posts?
‘PostGIS/Geopackage’
- Presenting drone imagery at a workshop? ‘MBTiles’
- A small dataset of greenspace areas in a neighborhood you
want to upload to the web? ‘GeoJSON’

LAB 4

GeoJSONs

Creating

geojson.io

Open Save New Meta

powered by Mapbox [Sign up for Mapbox](#)

Search

JSON

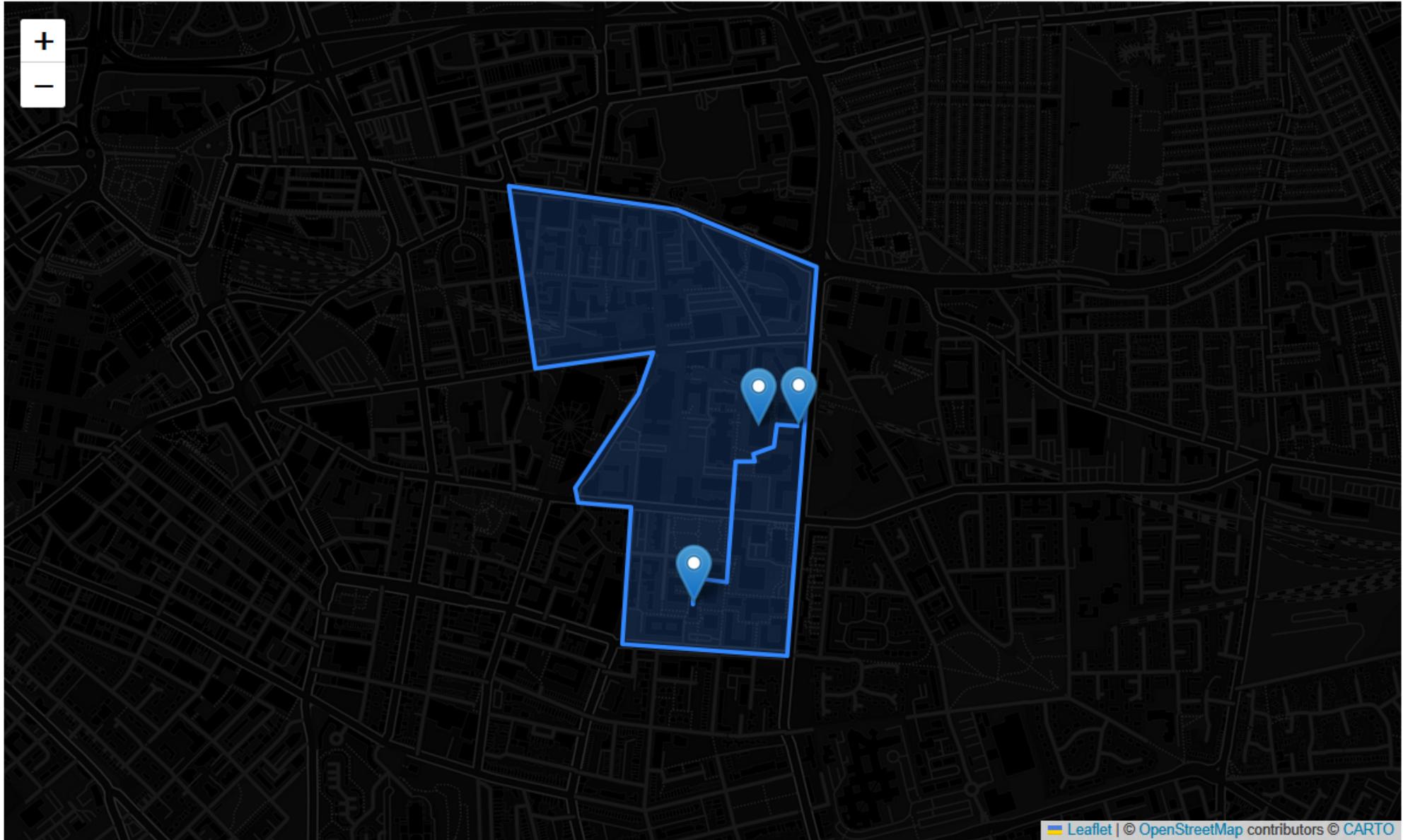
Table

Help

```
type: "FeatureCollection",
features: [
  {
    type: "Feature",
    properties: {},
    geometry: {
      coordinates: [
        -2.9958332123987645,
        53.40577976679657
      ],
      type: "Point"
    }
  }
]
```

geojson.io

Mapping



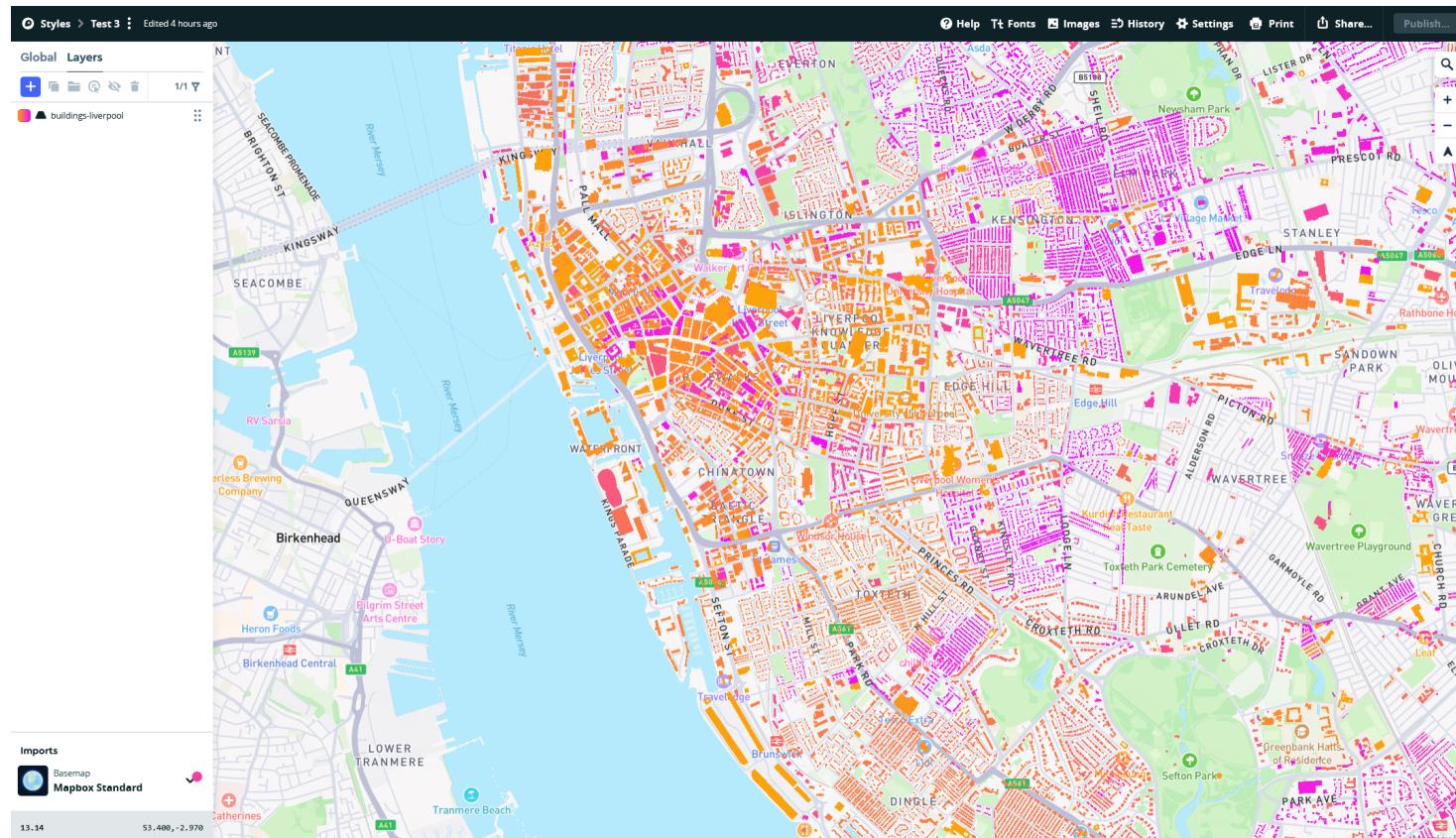
Leaflet | © OpenStreetMap contributors © CARTO

Mapbox and mbtiles

Creating .mbtiles and uploading

- Creating is **optional**, but if you can get it to work will greatly enhance your map.
- Upload either by the API (Again optional), or manually through the Mapbox Studio

Styles tool





Geographic Data Science by [Elisabetta Pietrostefani](#) is licensed under a [Creative Commons Attribution-ShareAlike 4.0 International License](#).