Engineer

GeoJSON

geography written as JSON

What it is

GeoJSON is an open format for exchanging map features, the points, lines and areas of a map together with the facts attached to them, as one plain JSON document. The IETF standardised it as RFC 7946 in August 2016, replacing the community specification first published in 2008. A GeoJSON text travels with the media type `application/geo+json` and is usually saved with the .geojson extension, although plain .json is registered as well.

The format has three kinds of object, and each one says which kind it is in a "type" member.

- **Geometry** is a shape. There are seven types: Point, MultiPoint, LineString, MultiLineString, Polygon, MultiPolygon and GeometryCollection. They are the shapes of the Simple Features model from the Open Geospatial Consortium (OGC), the same ones that the WKT and WKB encodings carry in text and binary.
- **Feature** is one thing on the map: a geometry, a "properties" object holding its attributes, and an optional "id". The geometry may be null for a feature whose location is unknown.
- **FeatureCollection** holds a list of features in a "features" member. It is what most files and API responses contain.

A single feature for a smart meter looks like this:

```
{
    "type": "Feature",
    "id": "meter-0417",
    "geometry": {
        "type": "Point",
        "coordinates": [12.5683, 55.6761]
```

```
    },
    "properties": {
        "name": "Meter 417",
        "installed": "2023-04-12"
    }
}
```

Each position is an array of numbers: longitude first, then latitude, then an optional altitude in metres. The specification says the first two come “precisely in that order”, so the point above sits in central Copenhagen, 12.57 degrees east and 55.68 degrees north. Every coordinate uses one reference system, WGS 84 longitude and latitude in decimal degrees, which the OGC identifies as CRS84. The 2008 version let a file name another system in a “crs” member. RFC 7946 removed that member because mixed reference systems had caused interoperability problems, and it allows another system only where every party has agreed on it in advance.

Polygons carry rules of their own. Each ring is closed, with at least four positions and the last one repeating the first. The first ring is the outer boundary and any later rings are holes. RFC 7946 adds a right-hand rule, with outer rings running counterclockwise and holes clockwise, but because the 2008 version said nothing about winding, parsers are told to accept polygons that break it.

Use the concept in a decision

Put “What is GeoJSON?” to work in a real decision. We can help relate the concept to the system you are building or buying. Anything that happens in a place — vessels, vehicles, plots, sensors — put on a map you can search, measure and share.

Explore our services Get in touch →

Test the assumptions behind the term

Use the explanation and linked work to frame questions for your project. We can examine the assumptions with you before a tool is chosen. Made a geographical data pipeline 50x faster — 50x faster geographical data pipelines

Made a geographical data pipeline 50x faster Get in touch →

Why it matters

Anything that reads JSON can read GeoJSON. A browser parses it with no plugin, a web-mapping library draws it directly and a document database stores it as it is. RFC 7946 notes that it “is widely used in JavaScript web-mapping libraries, JSON-based document databases, and web APIs”. That reach makes it the usual format whenever map data leaves a desktop GIS for a web page or an API.

Standards build on it as well. OGC API – Features, the OGC standard for serving geographic features over HTTP, recommends GeoJSON wherever the data can be represented in it. A server that offers GeoJSON returns a FeatureCollection for a list of features and a single Feature for one, both as `application/geo+json`, and its default reference system is the same WGS 84 longitude and latitude that GeoJSON fixes.

The same design choices set its limits.

- **One reference system.** Data held in a national grid has to be transformed into longitude and latitude before it is written, and transformed back by any consumer that measures in metres.
- **Simple shapes only.** The geometry types cover points, straight-sided lines and polygons. Curves, solids such as 3D city models and coordinates outside WGS 84 fall outside the format, and OGC API – Features gives exactly those cases as its examples of data that GeoJSON cannot carry.
- **Size.** Every coordinate is written out as decimal digits, and a FeatureCollection is a single JSON value, which an ordinary JSON parser reads to the last byte before handing back any feature in it. RFC 8142 defines GeoJSON Text Sequences, media type `application/geo+json-seq`, with one GeoJSON text per record, so that arbitrarily large datasets can be read incrementally.
- **No shared edges.** Two neighbouring polygons each store their common border in full, so the border is written twice, and an edit to one copy can open a gap or an overlap.

RFC 7946 is also plain about what the format leaves to others. GeoJSON provides no privacy or integrity protection of its own, so data in transit relies on TLS. The security section warns that publishing the locations of sensitive people, animals, habitats or facilities can expose them to tracking or harm, and that blurring positions in an anonymised dataset is unlikely to defeat a determined attack.

What it looks like in practice

GeoJSON was one of the formats that GDAL/OGR moved data between, alongside shapefiles, GeoTIFF, KML and vector tiles, on more than thirty GIS projects that were built, deployed and supported in production. PostgreSQL with PostGIS held the spatial data underneath, QGIS and JOSM served for the data work itself, and the browser maps on top used Mapbox GL and Leaflet.

A reference system has to be settled before any of that. In a migration from MSSQL to PostgreSQL with the PostGIS extension, over 50 million spatial records went through ETL pipelines whose transformation steps brought them to the SRID standard EPSG:4326. That code names the same WGS 84 datum in degrees that GeoJSON fixes, and the PostGIS manual states that the output of its ST_AsGeoJSON function conforms to RFC 7946 for geometry in EPSG:4326, with no transform needed.

Where it goes wrong

Points land in the wrong sea. A layer of Danish addresses appears in the middle of the Arabian Sea, and every range check still passes because both numbers are valid degrees. Latitude and longitude were swapped on the way in. EPSG:4326 formally lists latitude before longitude, many data sources follow it, and GeoJSON wants longitude first. Build each position from named fields rather than from column order, and keep a test feature at a known address so that a swap fails a test instead of a map review.

Coordinates arrive in metres. A layer is missing from the map or rejected by the client, and the file holds coordinates in the hundreds of thousands where degrees belong. The geometry was stored in a projected national grid and exported as it was. Transform it to longitude and latitude first, with `ST_Transform(geom, 4326)` in PostGIS. Left untransformed, ST_AsGeoJSON attaches a “crs” member by default, the member RFC 7946 removed.

Polygons draw inside out in one client. A service area fills the whole map except its own shape, while other tools draw the same file correctly. That client relies on the direction of each ring to tell inside from outside, and the rings run the other way, which the 2008 version allowed. The PostGIS manual notes that some clients require the right-hand rule, so apply it before export with `ST_ForcePolygonCCW(geom)`, and test the output in every client that consumes it.

Files carry more digits than the map needs. A layer of detailed polygons takes seconds to download on a phone, and its coordinates run to nine or more decimal places. ST_AsGeoJSON writes up to nine decimal places by default, and RFC 7946 notes that going from six to fifteen can nearly double a text’s size, while six already resolves about 10 centimetres. Pass a smaller `maxdecimaldigits`, and reduce precision with `ST_ReducePrecision` first, because the PostGIS manual warns that trimming digits on output alone can make a geometry invalid.

Internal columns ship inside properties. A public endpoint returns owner names, cost figures or notes that were meant to stay in the database. ST_AsGeoJSON, given a whole row, puts every column except the geometry into “properties”. Select only the columns meant for publication in a subquery, and, on PostGIS 3.5 or later, name the key column so that it becomes the feature’s “id”:

```
SELECT json_build_object(
  'type', 'FeatureCollection',
  'features', json_agg(ST_AsGeoJSON(m.*, id_column => 'id')::json)
)
FROM (SELECT id, name, geom FROM meters) AS m;
```

Further reading

- RFC 7946: The GeoJSON Format — the IETF standard, with the object types, coordinate order, winding rule, precision advice and security considerations.
- RFC 8142: GeoJSON Text Sequences — the companion format that splits a large dataset into one GeoJSON text per record.
- ST_AsGeoJSON — the PostGIS manual page for exporting geometry and whole rows as GeoJSON, with its precision and option flags.
- EPSG:4326 WGS 84 — the registry entry whose axis order puts latitude first, the opposite of GeoJSON.
- OGC API – Features – Part 1: Core — the OGC standard that recommends GeoJSON for serving features over HTTP.

Glossary

Choose for your actual constraints

Describe the workflow, the existing interfaces and the users who need the change. We can define a testable delivery and a handover that lets your team maintain it.

See what it costs Get in touch →

Make the terminology practical

Ask how “What is GeoJSON?” applies to your organisation. A concrete example from your current system is enough to start the discussion.

Explore our services

welcome@engineer.company +45 50 10 61 84 Copenhagen, Denmark

Take this page with you

Contact card

Carefully Crafted With Love by D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Copyright © 2025 – present · Engineer ApS · All rights reserved.

Printed from https://engineer.company/glossary/geojson/