**MapBox Vector Tile Structure**

# **mapbox-vector-tile-cs**

## VectorTile:

Get vector tile layers (Dictionary<string, byte[]>) name and data by method.

## VectorTileLayer:

|  |  |  |
| --- | --- | --- |
| Name | Property/Method | Comments |
| Data | Property | byte[] |
| Name | Property | string |
| Version | Property | ulong |
| Extent | Property | ulong |
| Keys | Property | List<object> |
| Values | Property | List<object> |
| FeatureCount() | Method | Get VectorTileFeature by feature index. |
| GetFeature(…) | Method | Get features count by method. |

## VectorTileFeature:

|  |  |  |
| --- | --- | --- |
| Name | Property/Method | Comments |
| Id | Property | ulong |
| Layer | Property | VectorTileLayer |
| GeometryType | Property | GeoType(UNKNOWN, POINT, LINESTRING, POLYGON) |
| GeometryCommands | Property | List<uint> |
| Tags | Property | List<int> |
| GetProperties() | Method | Get feature properties(Dictionary<string, object>) by method |

# **mapbox-vector-tile-js**

## VectorTile:

|  |  |  |
| --- | --- | --- |
| Name | Property/Method | Comments |
| layers | Property | Object — an object containing parsed layers in the form of {<name>: <layer>, ...}, where each layer is a VectorTileLayer object. |

## VectorTileLayer:

|  |  |  |
| --- | --- | --- |
| Name | Property/Method | Comments |
| name | Property | String |
| version | Property | Number — default: 1 |
| extent | Property | Number — default: 4096, tile extent size. |
| length | Property | Number — number of features in the layer |
| Feature(…) | Method | Get a feature (VectorTileFeature) by the given index from the layer. |

## VectorTileFeature:

|  |  |  |
| --- | --- | --- |
| Name | Property/Method | Comments |
| type | Property | Number — type of the feature (also see VectorTileFeature.types) |
| id | Property | Number — feature identifier, if present |
| extent | Property | Number — feature extent size. |
| properties | Property | Object — object literal with feature properties. |

## serialtiles-spec

1. **Header**: The stream MUST begin with the literal text. JSONBREAKFASTTIME
2. **TileJSON**: The second item in the stream MUST be a valid TileJSON object, stringified and wrapped in a JSON object under the member tilejson
3. **Tiles**: Each tile is a JSON object that contains z, x, y, and buffer members. The z x and y members represent a Z/X/Y tile in the slippy map tilenames scheme, in the Web Mercator map projection.
4. **Examples**:
   1. Vector Tiles (vector tile data):

JSONBREAKFASTTIME

{"tilejson":"info object"}

{"z":0,"x":0,"y":0,"buffer":"base64-encoded, gzipped tile data"}

{"z":1,"x":0,"y":0,"buffer":"base64-encoded, gzipped tile data"}

{"z":1,"x":1,"y":0,"buffer":"base64-encoded, gzipped tile data"}

{"z":1,"x":0,"y":1,"buffer":"base64-encoded, gzipped tile data"}

{"z":1,"x":1,"y":1,"buffer":"base64-encoded, gzipped tile data"}

* 1. Raster Images (png, webp, gif, jpg):

JSONBREAKFASTTIME

{"tilejson":"info object"}

{"z":0,"x":0,"y":0,"buffer":"base64-encoded data"}

{"z":1,"x":0,"y":0,"buffer":"base64-encoded data"}

{"z":1,"x":1,"y":0,"buffer":"base64-encoded data"}

{"z":1,"x":0,"y":1,"buffer":"base64-encoded data"}

{"z":1,"x":1,"y":1,"buffer":"base64-encoded data"}