

JSGeofencing

A geofencing extension for GeoJson

# Abstract

This specification defines a data model and JSON representation of geofence data that can be used for alerts in a motion and GPS navigation environment. It aims to be an alternative to custom data format and to be unambiguous, extendable and simple to process.

## Copyright notice

Copyright (c) 2021 DotVision and the persons identified as the document authors. All rights reserved.

Table of content

[Abstract 1](#_Toc69132900)

[Copyright notice 1](#_Toc69132901)

[Introduction 4](#_Toc69132902)

[Notational convention 4](#_Toc69132903)

[Relation to GeoJSON 4](#_Toc69132904)

[Structure of JSGeofencing objects 5](#_Toc69132905)

[Type signatures 5](#_Toc69132906)

[Data Types 6](#_Toc69132907)

[UTCDate 6](#_Toc69132908)

[LocalDate 6](#_Toc69132909)

[Duration 6](#_Toc69132910)

[Display string localization 6](#_Toc69132911)

[Common JSGeofencing properties 7](#_Toc69132912)

[@type 7](#_Toc69132913)

[uid 7](#_Toc69132914)

[enabled 7](#_Toc69132915)

[consumed 8](#_Toc69132916)

[displayName 8](#_Toc69132917)

[description 8](#_Toc69132918)

[tags 8](#_Toc69132919)

[modifiers 8](#_Toc69132920)

[Geofence Primitive 9](#_Toc69132921)

[Properties 9](#_Toc69132922)

[radius 9](#_Toc69132923)

[elevation 9](#_Toc69132924)

[priority 10](#_Toc69132925)

[alerts 10](#_Toc69132926)

[Alerts 10](#_Toc69132927)

[Scope 11](#_Toc69132928)

[Properties 11](#_Toc69132929)

[relativeTo 11](#_Toc69132930)

[category 11](#_Toc69132931)

[severity 12](#_Toc69132932)

[message 12](#_Toc69132933)

[Modifier 13](#_Toc69132934)

[Commons Properties 13](#_Toc69132935)

[scope 13](#_Toc69132936)

[category 13](#_Toc69132937)

[priority 13](#_Toc69132938)

[Modifier objects 14](#_Toc69132939)

[geoExpiration 14](#_Toc69132940)

[geoValidityPeriods 14](#_Toc69132941)

# Introduction

## Notational convention

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [[rfc2119]](https://tools.ietf.org/html/rfc2119).

The underlying format used for this specification is JSON. Consequently, the terms "object" and "array" as well as the four primitive types (strings, numbers, booleans, and null) are to be interpreted as described in Section 1 of[[rfc8259]](https://tools.ietf.org/html/rfc8259).

Some examples in this document contain "partial" JSON documents used for illustrative purposes. In these examples, three periods "..." are used to indicate a portion of the document that has been removed for compactness.

## Relation to GeoJSON

GeoJSON [[rfc7946](https://tools.ietf.org/html/rfc7946)] is an open standard format designed for representing simple geographical features, along with their non-spatial attributes. It is based on the JSON format. The data for geofence or set of geofences is represented by Feature Object and FeatureCollection Object in the GeoJSON format. In Addition to it:

* The GeoJSON Object type can be a Feature Object or a FeatureCollection Object.
* The Geometry Object type can be a Point, MultiPoint, LineString, MultiLineString, Polygon, MultiPolygon, and GeometryCollection.
* All feature properties should contain a geofencingExtension attribute, which is used for carrying the geofence informations.
* Styling is **NOT** part of this specification and MAY be carry by another property of the geoJSON. Some tools already process such style within the properties attribute. For example, [[leaflet](https://leafletjs.com/reference-1.7.1.html#geojson)] use this approach and we MAY relay on few properties :
  + Width
  + LineColor
  + LineAlpha
  + FillColor
  + FillAlpha

Because these extensions are part of the GeoJSON specification, back end implementation **SHOULD** simply ignore it.

For example, application **MAY** add a specific tag to the GeoJson properties such

"properties": {

"style": {

"color": "#02B8FA",

"width": 3

},

…

}

* All coordinate values are represented as [longitude, latitude, elevation?] defined in **WGS84**. Optional elevation is expressed as meter above Sea Level.
* For each feature, which contains MultiPoint, MultiLineString, MultiPolygon, or GeometryCollection, the geofencingExt is applied to **ALL** the elements.
* GeoJSON is 2D by design. The Geofence MAY represent also 3D volumes such Cylinder, Box or even Mesh. In this last case, 3D format may relay on standard mesh format such Obj or even GLTF2.0

Following is a sample request body for a geofence represented as a circle geofence geometry in GeoJSON using a center point and a radius

{

"type": "Feature",

"geometry": {

"type": "Point",

"coordinates": [48.64003107385474, 2.574231]

},

"properties": {

“geofenceExtension”: {

….

“@type”:”geoCircle”,

"radius": 500,

…

}

}

# Structure of JSGeofencing objects

A JSGeofencing object is a JSON object, which MUST be valid I-JSON (a stricter subset of JSON), as specified in  [[rfc8259]](https://tools.ietf.org/html/rfc8259). Property names and values are case-sensitive.

The object has a collection of properties, as specified in the following sections. Unless otherwise specified, all properties are optional; omitted properties MUST be treated identically to if that property had the value of null, unless otherwise specified.

## Type signatures

Types signatures are given for all JSON objects in this document. The following conventions are used:

* Foo: Any name that is not a native JSON type means an object for which the properties (and their types) are defined elsewhere within this document.
* Foo[]: An array of objects of type Foo.

## Data Types

In addition to the standard JSON data types, the following data types are used in this specification:

### UTCDate

This is a string in  [[rfc3339]](https://tools.ietf.org/html/rfc3339) date-time format, with the further restrictions that any letters MUST be in upper-case, the time component MUST be included and the time MUST be in UTC. Fractional second values MUST NOT be included unless non-zero and MUST NOT have trailing zeros, to ensure there is only a single representation for each date-time.

For example 2010-10-10T10:10:10.003Z is OK, but 2010-10-10T10:10:10.000Z is invalid and MUST be encoded as 2010-10-10T10:10:10Z.

In common notation, it should be of the form YYYY-MM-DDTHH:MM:SSZ.

### LocalDate

This is a date-time string with no time-zone/offset information. It is otherwise in the same format as UTCDate: YYYY-MM-DDTHH:MM:SS. The time-zone to associate the LocalDate with comes from an associated property, or if no time-zone is associated it defines floating time. Floating date-times are not tied to any specific time-zone. Instead, they occur in every timezone at the same wall-clock time (as opposed to the same instant point in time).

### Duration

A duration is represented by a subset of ISO8601 duration format, as specified by the following ABNF:

dur-secfrac = "." 1\*DIGIT

dur-second = 1\*DIGIT [dur-secfrac] "S"

dur-minute = 1\*DIGIT "M" [dur-second]

dur-hour = 1\*DIGIT "H" [dur-minute]

dur-time = "T" (dur-hour / dur-minute / dur-second)

dur-day = 1\*DIGIT "D"

duration = "P" (dur-day [dur-time] / dur-time)

In addition, the duration MUST NOT include fractional second values unless the fraction is non-zero. A zero duration MUST be represented as “P0D”.

## Display string localization

Some string properties in models are meant for display and, therefore, support localization. JSGeofencing models use JSON-LD's string internationalization support for localization. Each localizable property (e.g. displayName and description) is defined to be a JSON-LD language map ("@container": "@language"). The keys of the language map must be language tag strings (see [BCP 47](https://tools.ietf.org/html/bcp47)). [ISO 639](https://www.loc.gov/standards/iso639-2/php/code_list.php) provides a list of language tags. The default language for JSGeofencing documents is English.

# Common JSGeofencing properties

This section describes the properties that are common to the various JSGeofencing object types. Specific JSGeofencing object types may only support a subset of these properties.

### @type

Type: String

Required : Yes

Specifies the type which this object represents.

This **MUST** be one of the following values, or a vendor-specific value:

* Geometries
  + geoCircle
  + geoRect
  + geoArea
  + geoCylinder
  + geoBox
  + geoMesh
  + geoSegment
  + geoPath
* Modifiers
  + geoExpiration
  + geoValidityPeriods
  + geoOccurence
  + geoDelay
  + geoListener
* Alerts
  + geoAlert

A valid JSGeofencing object MUST include this property.

### uid

Type: String

Required : Yes

A globally unique identifier, used to associate the object as the same across different systems, geofencing models and views. The value of this property MUST be unique across **ALL** JSGeofencing objects, even if they are of different type. [[RFC4122]](https://tools.ietf.org/html/rfc4122) describes a range of established algorithms to generate universally unique identifiers (UUID), and the random or pseudo-random version is recommended to use.

A valid JSGeofencing object MUST include this property.

### enabled

Type : boolean

Required : No

Default : true

Define if the object is active or not. ie : a disabled object will not currently take part of the geofencing pipeline flow session

### consumed

Type : boolean

Required : No

Default : false

Define if the object is terminated ie : a consumed object will no longer take part of the geofencing pipeline flow current session

### displayName

Type : string

Required : No

Default : Null

Localizable : yes

A localizable name for display

### description

Type : string

Required : No

Default : Null

Localizable : yes

A localizable description for display

### tags

Type : string[]

Required : No

Default : Null

A set of string for miscellaneous purpose such search or filtering

### modifiers

Type: Modifier[]

Required : no

A set of modifier to be applied to the current object. Note that some modifier may have non sense for specific object. This specification do NOT cover this aspect.

# Geofence Primitive

Geofence node are the geofencing node relative to geometries. This include but not limited to type of

* geoCircle
* geoRect
* geoArea
* geoCylinder
* geoBox
* geoMesh
* geoSegment
* geoPath

## Properties

### radius

Type : number

Required : No

Default : 0

Nodes with underlying geometry of type Point, MultiPoint, LineString, MultiLineString **MAY** contain radius in properties. radius value is measured in meters, the radius value ranges from 1 to 10000.

Nodes with underlying geometry of type Polygon and MultiPolygon type does **NOT** have a radius property.

### elevation

Type : range

Required : No

Default : null

Nodes **MAY** contain elevation range in properties. This range act as extrusion of the underlying geometry. For example, a circle MAY become a cylinder.

Range values are measured in meters and are expressed as meter above or below Sea Level. A negative value indicate below Sea Level.

The optional elevations of the underlying geometries are not take in account.

A range object has the following properties

* min : number
* max: number

### priority

Type : number

Required : No

Default : 0

Some geometries may overlap. And in this case sorting the geometries with a priority value **MAY** be useful.

priority can range between 0 (lowest priority) and 127 (highest priority).

### alerts

Type: Alert[]

Required : no

Default : null

Set of alerts configuration for the current primitive.

*Note that each Primitive has its own set of Alert and has a default behavior.*

# Alerts

Geofencing pipeline is defined as Location flow associated with an user.

At each new user location, this new location and eventually segments of two consecutive location are processed against primitives.

Depending the result, one or several alerts MAY be raised.

This alerts are defined according the type of Geofencing primitive.

* Motion
  + speed
  + move
  + idle
* Geography
  + Commons alerts
    - distance
    - altitude
  + Fence Alerts
    - crossing
  + Zone Alerts
    - enter
    - exit
    - inside
    - outside

Note that this alerts **MAY** be altered by modifiers.

*Note that each Alert has its own set of Modifiers and has a default behavior.*

## Scope

* Alert’s definition serve the sole objective of this specification. The mechanism AND/OR format to transmit these alerts is out of scope of this specification. Event format **MUST** be defined through external specification such alarm and telemetry.
* It is common to merge threshold Alert within Geofence. For example a system MAY be interested by speed limit or to identify asset immobility. In term of geofencing, this threshold alert take place within geometry entities. Sole threshold alerts **MUST** be defined as broader scope such alarm and telemetry.

## Properties

### relativeTo

Type : string

Required : yes

The type of the event that the alarm is related to. Values are of :

* Motion
  + speed
  + move
  + idle
* Geography
  + altitude
  + distance
  + crossing
  + enter
  + exit
  + inside
  + outside

A valid Alert object MUST include this property.

### category

Type : string

Required : yes

Default : geography

The category to which this alert belongs. Alert Categories define groupings of alerts supported by an Geofencing Event server

Standard categories are:

* motion
* geography

Vendors **MAY** add categories, which **MUST** be prefixed with vendor namespace

<namespace>:<category>

### severity

Type : number

Required : No

Default : 1

The urgency of the event. This may be a value in the range of 1 – 1000, with 1 being the lowest severity and 1000 being the highest. Typically, a severity of 1 would indicate in event which is informational in nature, while a value of 1000 would indicate an event of catastrophic nature which could potentially result in severe financial loss or loss of life. It is expected that few server implementations will support 1000 distinct severity levels. Therefore, server developers are responsible for distributing their severity levels across the 1 – 1000 range in such a manner that clients can assume a linear distribution. n. For example, a client wishing to present five severity levels to a user could implement the severity level as shown in the following Figure

|  |  |
| --- | --- |
| Severity | JSGeofencing Severity |
| HIGH | 801 – 1000 |
| MEDIUM HIGH | 601 – 800 |
| MEDIUM | 401 – 600 |
| MEDIUM LOW | 201 – 400 |
| LOW | 1 – 200 |

### message

Type : string

Required : No

Default : Null

Localizable : yes

A localizable message for display

# Modifier

JSGeofencing object are extensible using the concept of modifier. A modifier is an object which when applied to another object change his properties and/or behavior.

Every JSGeofencing object **MAY** have one or more modifier.

## Commons Properties

### scope

Type : string

Required : No

Default : “asset”

The scope of the modifier. Value can be :

* static : the modifier act as global level and any status/action are related to the model. For example, we can invalidate a zone if ANY of user exit it.
* asset : the modifier act as user level and any status/action are related and specific to this user. For example, a count modifier will store different information for different user and will act accordingly.

### category

Type : string

Required : No

Default : null

The category to which this modifier belongs. Modifier Categories define groupings of modifier supported by Geofencing tools.

Standard Supported value are, but not limited to :

* Calendar
* Filter
* Geography
* Time
* Math
* Geometry
* Motion
* Environmental

### priority

Type : number

Required : No

Default : 0

Some modifiers may be call prior to other.

priority can range between 0 (lowest priority) and 127 (highest priority).

## Modifier objects

### geoExpiration

Define the expiration of the underlying node.

#### Properties

##### when

Type : UTCDate | LocalDate

Required : yes

The absolute expiration date time.

Once a Node is expired, the consumed property is set to true.

A valid modExpiration object MUST include this property.

### geoValidityPeriods

Define the validity periods of the underlying node.

#### properties

##### periods

Type : Period[]

Required : yes

An Period object has the following properties

* From: UTCDate | LocalDate
* to: UTCDate | LocalDate
* frequency: String This MUST be one of the following values:
  + yearly
  + monthly
  + weekly
  + daily
  + hourly
  + minutely
  + secondly
* interval: Number(optional, defaults to 1)
* count: Number(optional). This MUST NOT be included if an until property is specified.
* until: UTCDate | LocalDate (optional). This MUST NOT be included if a count property is specified.

A valid modVailityPeriods object MUST include this property.