Observation JSON Format

Présentation

06/01/2022

Environmental Sensing

Table of Contents

[1 Introduction 2](#_Toc92572616)

[1.1 Conventions used 2](#_Toc92572617)

[1.2 Terminology 2](#_Toc92572618)

[1.3 Rules 2](#_Toc92572619)

[2 ObsJSON objects 3](#_Toc92572620)

[2.1 ESAtt 3](#_Toc92572621)

[2.2 ESObservation 3](#_Toc92572622)

[2.3 ESObs 4](#_Toc92572623)

[2.4 ESValue 4](#_Toc92572624)

[2.4.1 LocationValue 5](#_Toc92572625)

[2.4.2 DatationValue 6](#_Toc92572626)

[2.4.3 PropertyValue 6](#_Toc92572627)

[2.4.4 ResultValue 7](#_Toc92572628)

[2.5 ESData 7](#_Toc92572629)

[2.5.1 ESInformation 8](#_Toc92572630)

[2.5.2 ESParameter 8](#_Toc92572631)

# Introduction

ObsJSON is a text format for the ES-Observation data.

This format is an application of the JSON format (RFC 8259), GeoJSON format (RFC 7946), Date and Time format (RFC 3339).

## Conventions used

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

The grammatical rules in this document are to be interpreted as described in [RFC5234].

## Terminology

The terms Json-Text, Json-Value (Value), Object, Member, Array, Number, String, False, Null, True are define in the JSON grammar.

The terms Geometry-type, Point, MultiPoint, LineString, MultiLineString, Polygon, MultiPolygon, GeometryCollection, GeoJSON-Types are defined in GeoJSON grammar.

Timestamp is defined in Date and Time format.

## Rules

A Value in an ESValue SHOULD be unambiguous (i.e., parsers CAN deduce the Value type).

The Array SHOULD NOT include user’s values.

The Objects MAY include users’ elements.

An ObsJSON-Text CAN contains all the ESObservation information (i.e., the ESObservation build from the ObsJSON-Text is identical to the initial ESObservation).

Values in Array are ordered and independent from the other Values.

# ObsJSON objects

An ObsJSON -Text is a JSON-text and consists of a single object ObsJSON

ObsJSON objects define in this document are:

* ESObservation
* ESObs (ESDatation, ESLocation, ESProperty, ESResult)
* ESValue (DatationValue, LocationValue, PropertyValue, ResultValue)
* ESData (ESInformation, ESParameter, ESUserData)
* ESAtt

## ESAtt

ESAtt are Members Key / Value included in an Object.

Two kind of ESAtt are defined:

* Defined ESAtt where the Key and the format of the Value are specified,
* User ESAtt where both Key and Value are unconstrained.

User ESAtt MAY be included in all Objects.

## ESObservation

**Description**

ESObservation is an Object. The Members included SHALL be:

* ESAtt,
* ESObs,
* ESData

The Defined ESAtt are:

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Member | Key | Value |
| ESAtt | Type | « type » | « observation » |
| ESAtt | Id | « id » | String |
| ESObs | ESDatation | « datation » | Array or DatationValue |
| ESObs | ESLocation | « location » | Array or LocationValue |
| ESObs | ESProperty | « property » | Array or PropertyValue |
| ESObs | ESResult | « result » | Array or ResultValue |
| ESData | ESInformation | « information » | Object |
| ESData | ESParameter | « parameter » | Object |
| ESData | ESUserData | « userdata » | Object |

**Validity**

An ESObservation is valid if:

* it contains at least the Type Member,
* each Member is valid.

**Example**

{“type”:“observation”, “datation”:“morning”, “location”:“paris”, “property”:“air quality”, “result”:“good” }

{“type”:“observation”, “datation”:“2021-01-05T22:18:26”, “location”:[2.4, 48.9], “property” : {“typep”:“PM10”, “unit”:“µg/m3”  }

## ESObs

**Description**

ESObs is a Value Array. Value is specific for each ESObs:

|  |  |  |
| --- | --- | --- |
| Member | Key | Value |
| ESDatation | « datation » | DatationValue |
| ESLocation | « location » | LocationValue |
| ESProperty | « property » | PropertyValue |
| ESResult | « result » | ResultValue |

*Note:*

*If the Array contains only one Value, the square brackets MAY be omitted.*

**Validity**

An ESObs Member is valid if:

* it contains at least a Value
* each Value is valid

**Example**

“location”:“paris”

“location”:[[2.4, 48.9], [4.8, 45.8], [5.4, 43.3]]

“location”:[“paris”, “lyon”, “marseille”]

## ESValue

**Description**

An ESValue contains two information: A Name, a Value

One of the three formats SHALL be used for ESValue (where Name is a String):

* Object format: { Name : Value }
* Value format: Value
* Name format: Name

**Validity**

An ESValue Member is valid if:

* One of the three formats is used,
* it contains at least a Value or a Name
* each Value is valid compared to ESObs

*Note: For ResultValue, the Name format is not allowed (i.e., if the ResultValue consists of a single string, this SHOULD be interpreted as a Value.*

**Example**

“morning” *Name format*

{“morning”: “2021-01-05T10:00:00”} *Object format*

[[“2021-01-05T08:00:00”, “2021-01-05T12:00:00”]] *Value format*

### LocationValue

**Description**

The Value is a representation of a Point or a Polygon. It is defined by a Coordinates Array (as specified in GeoJSON).

**Validity**

A Value is valid if the Coordinates Array is valid and represents a Point or a Polygon.

**Example**

[2.4, 48.9] *Point*

[[[2.4, 48.9], [4.8, 45.8], [5.4, 43.3], [2.4, 48.9]]] *Polygon*

[[[0,0], [0,5], [5,5], [0,0]], [[1,1], [1,2], [2,2], [1,1]]] *Polygon with a hole*

*Note:*

*The other Geometry-type are not allowed because the Coordinates Array is ambiguous:*

* *LineString, Multipoint and Array of Point have the same representation*
* *Polygon and Array of LineString have the same representation*
* *MultiPolygon and Array of Polygon have the same representation*

*The LineString in a Polygon MAY be open (without the last Point)*

### DatationValue

**Description**

A Date is defined by a Timestamp (as specified in RFC 3339).

The Value is a representation of a single Date or a Slot (MultiInterval):

* Date: String
* Slot: Array of one or multiple Interval (Array of two Date)

**Validity**

A Value is valid if the Date or Slot is valid.

**Example**

“2021-01-05T10:00:00” *Date*

[[“2021-01-05T08:00:00”, “2021-01-05T12:00:00”]] *Interval*

[[“2021-01-05”, “2021-01-10”], [“2021-01-20”, “2021-01-25”]] *Slot*

*Note:*

*Intervals MUST be represented by a Slot to avoid ambiguities with an array of Dates*

*If the DatationValue consists of a unique String, and if the String represents a Date, the parser SHALL assign the String to the Date, otherwise to the Name.*

### PropertyValue

**Description**

The Value is an Object made up of ESAtt. The Defined ESAtt are:

|  |  |  |
| --- | --- | --- |
| Member | Key | Value |
| PropertyType | « ptype » | String |
| Unit | « unit » | String |
| SamplingFunction | « sampling » | String |
| Application | « application » | String |
| SensorType | « sensor » | String |
| UpperValue | « uppervalue » | Float |
| LowerValue | « lowervalue » | Float |
| Period | « period » | Float |
| UpdateInterval | « updateinterval » | Float |
| Uncertainty | « uncertainty » | Float |

**Validity**

A Value is valid if it contains at least the PropertyType ESAtt.

**Example**

{“ptype”:“Temperature”} *Minimal Value*

{“ptype”:“Temperature”, “unit”:“°c”} *Defined Member*

{“ptype”:“Temperature”, “operation”:“phase 1”} *User Member*

### ResultValue

**Description**

The Value contains two information: A Result, an Index

* The Result SHALL be a numeric value (number), a string value or a literal value (null, true, false).
* The Index is an Array value composed with three integer numbers.

Two formats are defined for the Value:

* Result format: only the Result value
* Index format: Array with first the Result and second the Index

**Validity**

An Index is valid if the integer numbers are positive.

A Value is valid if it contains at least a Result and if the Index (if present) is valid.

**Example**

21.8 *Numeric Value*

[“low temperature”, [0,2,1]] *Index format*

“https://loco-philippe.github.io/ES.html” *String value*

*Note:*

*If the ResultValue is composed by a unique String, the parser SHALL assign the String to the Result.*

## ESData

ESData are elements where the Value is an Object.

The Object is made up of Defined ESAtt and User ESAtt and contains at least one Member.

### ESInformation

**Description**

The Defined ESAtt are:

|  |  |  |
| --- | --- | --- |
| Member | Key | Value |
| ObservationType | « typeobs » | String |
| LocationType | « typeloc » | String |
| DatationType | « typedat » | String |
| PropertyType | « typeprp » | String |
| ResultType | « typeres » | String |
| nValLocation | « nvalloc » | Integer |
| nValDatation | « nvaldat » | Integer |
| nValProperty | « nvalprp » | Integer |
| nValResult | « nvalres » | Integer |
| BoundingBox | « bbox » | Array (4 Float) |
| IntervalBox | « tbox » | Array (2 String) |
| Complet | « complet » | True / false |
| Score | « score » | Integer |
| MeasureRate | « measurerate » | Float |
| Dimension | « dimension » | Integer |
| Axes | « axes » | Array (1 to 3 integers) |

**Validity**

An ESInformation is valid if the Value contains at least one ESAtt.

All the ESAtt are optional.

**Example**

{"typeobs":"areaObsrecord"} *Minimal Value*

{"typeobs":"areaObsrecord", "complet":false, "score":226} *Defined Member*

*Note:*

*Those information come from the other ESObervation elements. A parser MAY ignore The ESInformation to build an ESObservation.*

### ESParameter

**Description**

The Defined ESAtt are:

|  |  |  |
| --- | --- | --- |
| Member | Key | Value |
| Reference | « reference » | String |
| ResultTime | « resulttime » | Timestamp |
| Order | « order » | Enum |
| UniqueIndex | « unicindex » | True/false |

**Validity**

An ESParameter is valid if the Value contains at least one ESAtt.

All the ESAtt are optional.

**Example**

{"order":"dlp", “unicindex”:true}