ES JSON Format

Présentation

01/02/2023

Environmental Sensing

Table of Contents

1. Introduction

The Environmental Sensing JSON format is described in three complementary documents:

- ESJSON for the part related to elementary values (ESValue)

- IlistJSON for the part related to indexed lists (Ilist and Iindex)

- ObsJSON for the full Observation object (Observation)

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

A binary version is also defined (Appendix) with CBOR format (RFC 8949)

This document specifies the ESJSON format.

1.1 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].

1.2 Terminology

The terms Json-Text, Json-Value (Value), Object, Member, Element, Array, Number, String, False, Null, True are defined 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.

1.3 Rules

An ESValue Json-Text SHOULD be unambiguous (i.e., parsers CAN deduce the ESValue object).

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

Members in Objects are not ordered.

2. Environmental Sensing – Data

2.1 Principles

The ESValue family is a set of name-value pairs objects according to the 3 dimensions:

- Temporal,
- Spatial,
- Physical (observed property)

Two complementary dimensions are added to include simple objects and complex objects.

2.2 ESValue

Five categories of ESValue (one Type for each) are defined:

- DatationValue
- LocationValue
- PropertyValue
- NamedValue
- ExternValue

The simple objects included in the NamedValue category are defined in Appendix.

The ExternValue category includes all values that are not in the other categories (i.e. complex objects).

2.3 Structure

Description

An ESValue is defined by a Type, a Name and a Value. The Type defines the category of ESValue. The Name is complementary information of the Value. The Value is a value that matches the defined categories.

Two Object formats are used for ESValue:

- TypedValue format (where Type is defined outsideTypedValue)
- UntypedValue format (where Type is defined inside UntypedValue}

Validity

An ESValue is valid if:

- One of the two formats is used,
- The TypedValue format is used in cases where the Type is known
- The TypedValue is valid for the Type

Note:

If the ESValue contains more than one Member, the TypedValue format is used.
If the key of the Object is not recognized as a valid Type, the TypedValue format is used.

If the Object contains one Member and the key is a valid Type, the UntypedValue format is used.

2.4 TypedValue

Description

One of the three formats SHALL be used for TypedValue :

- Name-Value format: {name: value}
- Value format: value
- Name format: name

Validity

A TypedValue is valid if:

- One of the three formats is used,
- it contains at least a value or a name
- value is valid for the Type
- it contains at least a value or a name
- value is valid for the Type
- name is a String

Note:

The name string MAY be used to represent:

- *detailed information (e.g., "beginning of the observation"),*
- *link to external information (e.g., "https://loco-philippe.github.io/ES.html"),*
- *id to link internal information (e.g., "res003" where "res003" is a key),*

If the TypedValue is an Object and contains more than one Member, the Value Format is used.

Example

"morning"	*Name format*
{"morning": "2021-01-05T10:00:00"}	*Name-Value format*
[["2021-01-05T08:00:00", "2021-01-05T12:00:00"]]	*Value format*

2.5 UntypedValue

Description

One format is used for UntypedValue : {type: typedvalue} where type is a String and typedvalue a TypedValue:

Validity

A UntypedValue is valid if:

- the format is used,
- the type is a valid Type (i.e. string present in the TypeCatalog (see Appendix)
- the typedvalue is a valid TypedValue
- the typedvalue is valid for the Type

Example

```
{"datation": "morning"}}

{"datation": {"morning": "2021-01-05T10:00:00"}}

{"datation": [["2021-01-05T08:00:00", "2021-01-05T12:00:00"]]}
```

3. DATATIONVALUE

Type : "datvalue" or "datation"

Description

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

The Value of a DatationValue is a representation of a single Date, an Interval (Array of two Dates) or a Slot (MultiInterval):

- Date: String
- Interval : Array of one Array of two Date
- Slot: Array of multiple Array of two Date

Validity

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

Value 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.

4. LocationValue

Type : "locvalue" or "location"

Description

The Value of a LocationValue 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.

Value 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 others 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)

5. PROPERTYVALUE

Type : "prpvalue" or "property"

Description

The Value of a PropertyValue is an Object. The Members are:

Member	Key	Value
PropertyType	« prp »	String (mandatory)
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
UserMember	String	JSON Object

Note:

If the PropertyValue consists of a single Member (the PropertyType), it's not allowed to replace the Object by a string (the PropertyType value).

If the PropertyDict is not defined, the default PropertyDict is used

Validity

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

The PropertyType value MAY be present in a PropertyDict how's define the Unit value

Value example

{"prp": "Temp"}	*Minimal Value*
{"prp": "Temp", "unit": "°c"}	*Defined Member*
{"prp": "Temp", "unit": "°c", "operation": "phase 1"}	*User Member*

Note:

For PropertyValue, the Name format is allowed (i.e., if the PropertyValue consists of a single string, this SHOULD be interpreted as the name).

6. NamedValue

Type : "namvalue"

Description

The Value of a NamedValue CAN be any Json-Value.

Note:

The TypedValue format is always used (i.e. UntypedValue format is not allowed).

The Name format is not allowed (i.e., if the Value consists of a single string, this SHOULD be interpreted as a Value).

Validity

Only the ESValue Rules.

Example

21.8	*Value format*
{"age": 25}	*Name-Value format*
{"coord": [2.4, 48.9]}	*Name-value format*
"test"	*Value format*

7. EXTERNVALUE

The Type of an ExternValue is the concatenation of the character '$' and the Value Type(i.e. the class of the Value).

The Value of an ExternValue is a value associated with none of the other ESValue categories (see Appendix).

An ExternValue SHOULD have a *freestanding* Json representation (i.e. the Json-Value is enough to rebuild the Value).

Type : **'$' +** type of the Value

Description

The Value of an ExternValue CAN be any Json-Value and SHOULD be consistent with the Type.

Validity

Only the ESValue Rules.
The conformity to the Type is extern

Note:

> *The UntypedValue format is always used if the context is unable to define the Type*
>
> *If TypedValue format is used, the parser MUST be able to decode the JSON Object defined by the Type-Value format, otherwise the parser decodes the JSON Object as a Value.*

1. APPENDIX: VALUE TYPOLOGY

Simple data:

Simple data corresponds to the following objects:

- Numeric (integer of float)
- String
- Boolean
- Simple data assemblies :
  - Dictionary (set of key-value pairs)
  - List or Array (ordered set of values)
- Date objects

ESValue Type:

Object	Type
Observation	obs
DatationValue	datvalue
LocationValue	locvalue
PropertyValue	prpvalue
NamedValue	namvalue
ExternValue	extvalue
Ilist	ilist
datetime	datetime
Iindex	iindex
TimeSlot	timeslot

The type of ExternValue is composed by the '$' character and the value type (e.g. $obs, $ilist).

2. APPENDIX : RESERVED VALUES

to complete