DRAFT INTERNATIONAL STANDARD

ISO/IEC DIS 8825-8

ISO/IEC JTC **1**/SC **6**

Secretariat: **KATS**

Voting begins on:
2020-09-28

Voting terminates on:
2020-12-21

Information technology — ASN.1 encoding rules —

Part 8:
Specification of JavaScript Object Notation Encoding Rules (JER)

ICS: 35.100.60

This document is circulated as received from the committee secretariat.

Reference number
ISO/IEC DIS 8825-8:2020(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

CONTENTS

Page

Page

Introduction

The publications Rec. ITU-T X.680 | ISO/IEC 8824-1, Rec. ITU-T X.681 | ISO/IEC 8824-2, Rec. ITU-T X.682 | ISO/IEC 8824-3, Rec. ITU-T X.683 | ISO/IEC 8824-4 together describe Abstract Syntax Notation One (ASN.1), a notation for the definition of messages to be exchanged between peer applications.

This Recommendation |International Standard defines encoding rules that may be applied to values of ASN.1 types defined using the notation specified in the above-mentioned publications. Application of these encoding rules produces a transfer syntax for such values. It is implicit in the specification of these encoding rules that they are also to be used for decoding.

There are more than one set of encoding rules that can be applied to values of ASN.1 types. This Recommendation | International Standard defines a set of JavaScript Object Notation Encoding Rules (JER), so called because the encodings they produce are instances of the JSON grammar specified in ECMA 404.

This Recommendation | International Standard specifies the syntax and semantics of JER encoding instructions which modify the JSON text produced by the application of JER to certain ASN.1 types.

Clauses 8 to 12 list the JER encoding instructions and specify the syntax for their assignment to an ASN.1 type or component using either a JER type prefix (see Rec. ITU-T X.680 | ISO/IEC 8824-1, clause 31.3) or a JER encoding control section (see Rec. ITU-T X.680 | ISO/IEC 8824-1, clause 54).

Clause 13 defines the order of precedence if JER encoding instructions are present in both a JER type prefix and in a JER encoding control section.

Clauses 14 to 19 specify:

a) the syntax of each JER encoding instruction used in a type prefix or a JER encoding control section;

b) restrictions on the JER encoding instructions that can be associated with a particular ASN.1 type (resulting from inheritance and multiple assignments).

Clauses 20 to 41 specify the JER encoding of ASN.1 types, referencing earlier clauses which define the JER encoding instructions.

Annex A is informative and contains examples of JER encodings where JER encoding instructions are not used.

Annex B is informative and contains examples of JER encoding instructions and their effect on the JER encodings.

INTERNATIONAL STANDARD
ITU-T RECOMMENDATION

Information technology – ASN.1 encoding rules: Specification of JavaScript Object Notation Encoding Rules (JER)

1 Scope

This Recommendation | International Standard specifies a set of JavaScript Object Notation Encoding Rules (JER) that may be used to derive a transfer syntax for values of types defined in Rec. ITU-T X.680 | ISO/IEC 8824-1, Rec. ITU-T X.681 | ISO/IEC 8824-2, Rec. ITU-T X.682 | ISO/IEC 8824-3, Rec. ITU-T X.683 | ISO/IEC 8824-4. It is implicit in the specification of these encoding rules that they are also to be used for decoding.

The encoding rules specified in this Recommendation | International Standard:

- are used at the time of communication;
- are intended for use in circumstances where interoperability with applications using JSON is the major concern in the choice of encoding rules;
- allow the extension of an abstract syntax by addition of extra values for all forms of extensibility described in Rec. ITU-T X.680 | ISO/IEC 8824-1.

This Recommendation | International Standard also specifies the syntax and semantics of JER encoding instructions, and the rules for their assignment and combination. JER encoding instructions can be used to control the JER encoding for specific ASN.1 types.

2 Normative references

The following Recommendations and International Standards contain provisions which, through reference in this text, constitute provisions of this Recommendation | International Standard. At the time of publication, the editions indicated were valid. All Recommendations and Standards are subject to revision, and parties to agreements based on this Recommendation | International Standard are encouraged to investigate the possibility of applying the most recent edition of the Recommendations and Standards listed below. Members of IEC and ISO maintain registers of currently valid International Standards. The Telecommunication Standardization Bureau of the ITU maintains a list of currently valid ITU-T Recommendations.

2.1 Identical Recommendations | International Standards

- Recommendation ITU-T X.680 (2020) | ISO/IEC 8824-1:2020, *Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation*.
- Recommendation ITU-T X.681 (2020) | ISO/IEC 8824-2:2020, *Information technology – Abstract Syntax Notation One (ASN.1): Information object specification*.
- Recommendation ITU-T X.682 (2020) | ISO/IEC 8824-3:2020, *Information technology – Abstract Syntax Notation One (ASN.1): Constraint specification.*
- Recommendation ITU-T X.683 (2020) | ISO/IEC 8824-4:2020, *Information technology – Abstract Syntax Notation One (ASN.1): Parameterization of ASN.1 specifications*.
- Recommendation ITU-T X.690 (2020) | ISO/IEC 8825-1:2020, *Information technology – ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)*.
- Recommendation ITU-T X.691 (2020) | ISO/IEC 8825-2:2020, *Information technology – ASN.1 encoding rules: Specification of Packed Encoding Rules (PER)*.

2.2 Additional references

- ECMA Standard ECMA-404 (2013) – *The JSON Data Interchange Format*.
- IETF RFC 2045 (1996), *Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies.*
- *ISO International Register of Coded Character Sets to be Used with Escape Sequences.*
- ISO/IEC 10646:2003, *Information technology – Universal Multiple-Octet Coded Character Set (UCS).*

3 Definitions

For the purposes of this Recommendation | International Standard, the following definitions apply:

3.1 Specification of Basic Notation

For the purposes of this Recommendation | International Standard, all the definitions in Rec. ITU-T X.680 | ISO/IEC 8824-1 apply.

3.2 Information Object Specification

For the purposes of this Recommendation | International Standard, all the definitions in Rec. ITU-T X.681 | ISO/IEC 8824-2 apply.

3.3 Constraint Specification

This Recommendation | International Standard makes use of the following terms defined in Rec. ITU-T X.682 | ISO/IEC 8824-3:

a) component relation constraint;

b) table constraint.

3.4 Parameterization of ASN.1 Specification

This Recommendation | International Standard makes use of the following term defined in Rec. ITU-T X.683 | ISO/IEC8824-4:

– variable constraint.

3.5 Basic Encoding Rules (BER)

This Recommendation | International Standard makes use of the following terms defined in Rec. ITU-T X.690 | ISO/IEC 8825-1:

a) data value;

b) dynamic conformance;

c) encoding (of a data value);

d) receiver;

e) sender;

f) static conformance.

3.6 Packed Encoding Rules (PER)

This Recommendation | International Standard makes use of the following terms defined in Rec. ITU-T X.691 | ISO/IEC 8825-2:

a) composite type;

b) composite value;

c) outermost type;

d) relay-safe encoding;

e) simple type;

f) textually dependent.

3.7 Additional definitions

3.7.1 abstract syntax value: A value of an abstract syntax (defined as a set of values of a single ASN.1 type) which is to be encoded by JER or which is generated by JER decoding.

3.7.2 associated encoding instruction (for a type): A set of JER encoding instructions associated with a type.

3.7.3 effective value constraint (of an integer type): The smallest integer range that includes all the values of the integer type that are permitted by the JER-visible constraints (see clause 7.2.7).

3.7.4 effective size constraint (of a bitstring type): The smallest integer range that includes the lengths of all the values of the string type that are permitted by the JER-visible constraints (see clause 7.2.8).

3.7.5 final encoding instructions (for a type): The set of JER encoding instructions associated with a type as a result of the complete ASN.1 specification, and which are applied in producing encodings of that type.

3.7.6 inherited encoding instructions: A set of JER encoding instructions that are associated with the type identified by a type reference.

3.7.7 JSON array: A series of JSON tokens that constitute an array structure as specified in ECMA 404, clause 7.

3.7.8 JSON member name string (of a component of a sequence, set, or choice type that is encoded as a JSON object): The Unicode character string denoted by the name of the member of the JSON object identifying the component in the JER encoding.

3.7.9 JSON number: A JSON token that is a number as specified in ECMA 404, clause 8.

3.7.10 JSON object: A series of JSON tokens that constitute an object structure as specified in ECMA 404, clause 6.

3.7.11 JSON string: A JSON token that is a string as specified in ECMA 404, clause 9.

NOTE – A JSON string is part of a JER encoding, it begins and ends with a quotation mark, may contain escapes, and is distinct from the Unicode character string that it denotes.

3.7.12 JSON token: A Unicode character string that is one of the several kinds of tokens specified in ECMA 404, clause 4.

3.7.13 JER encoding instruction: Notation used to change the JER encoding of a type.

3.7.14 JER-visible constraint: An instance of use of the ASN.1 constraint notation that affects the JER encoding of a value.

3.7.15 octet: A group of eight consecutive bits, numbered from bit 8 (the most significant bit) to bit 1 (the least significant bit).

3.7.16 prefixed encoding instruction: A JER encoding instruction that is assigned to a type using a type prefix.

NOTE – Prefixed encoding instructions can delete, replace, or add to the associated encoding instructions of a type.

3.7.17 targeted encoding instruction: A JER encoding instruction that is assigned to multiple types using a target list in a JER encoding control section.

NOTE – Targeted encoding instructions can delete, replace, or add to the associated encoding instructions of multiple types.

4 Abbreviations

For the purposes of this Recommendation | International Standard, the following abbreviations apply:

ASN.1	Abstract Syntax Notation One
BER	Basic Encoding Rules of ASN.1
JER	JavaScript Object Notation Encoding Rules of ASN.1
PER	Packed Encoding Rules of ASN.1
PDU	Protocol Data Unit

5 Encodings specified by this Recommendation | International Standard

5.1 This Recommendation | International Standard specifies a set of encoding rules which can be used to encode and decode the values of an abstract syntax defined as the values of a single (known) ASN.1 type. This clause describes their applicability and properties.

5.2 JER encodings are always relay-safe provided the abstract values of the types **`EXTERNAL`**, **`EMBEDDED PDV`**, and **`CHARACTER STRING`** are constrained to prevent the carriage of OSI presentation context identifiers.

5.3 If a type encoded with JER contains **`EXTERNAL`**, **`EMBEDDED PDV`**, or **`CHARACTER STRING`** types, then the outer encoding ceases to be relay-safe unless the transfer syntax used for all the **`EXTERNAL`**, **`EMBEDDED PDV`**, or **`CHARACTER STRING`** types is relay-safe.

NOTE – The character transfer syntaxes supporting all character abstract syntaxes of the form **`{iso standard 10646 level-1(1) …}`** are canonical. Those supporting **`{iso standard 10646 level-2(2) …}`** and **`{iso standard 10646 level-3(3) …}`** are not always canonical. All the above character transfer syntaxes are relay-safe.

5.4 JER encodings are self-delimiting. Encodings are always a whole multiple of eight bits. When carried in an **`EXTERNAL`** type, they shall be carried in the **`OCTET STRING`** choice alternative, unless the **`EXTERNAL`** type itself is encoded in JER, in which case the value may be encoded as a single ASN.1 type (i.e., an open type). When carried in an

OSI presentation protocol, the "full encoding" (as defined in Rec. ITU-T X.226 | ISO/IEC 8823-1) with the **OCTET STRING** alternative shall be used.

5.5 This Recommendation | International Standard also specifies the syntax and semantics of JER encoding instructions (see clauses 14 to 19).

5.6 ASN.1 forms a basic JSON schema notation. The ASN.1 schema is used to define the content and structure of data using ASN.1 and the JavaScript Object Notation Encoding Rules. It can be used without JER encoding instructions.

5.7 JER encoding instructions provide wider flexibility in the JSON texts that can be specified.

5.8 JER encoding instructions are assigned to ASN.1 type definitions or to type references using either or both of JER type prefixes (see Rec. ITU-T X.680 | ISO/IEC 8824-1, clause 31.3) and a JER encoding control section (see Rec. ITU-T X.680 | ISO/IEC 8824-1, clause 54). If encoding instructions are associated with a type definition, they are carried with the ASN.1 type (through its type reference) into other type definitions and other ASN.1 modules. The final encoding instructions of a type are applied when the type is encoded in JER and modify the JSON text produced.

6 Conformance

6.1 Dynamic conformance for the JavaScript Object Notation Encoding Rules is specified in clauses 7 to 41.

6.2 Static conformance is specified by those standards which specify the application of these encoding rules.

6.3 Alternative encodings are permitted by the JavaScript Object Notation Encoding Rules as encoder's options. Decoders that claim conformance to JER shall support all JER encoding alternatives.

6.4 The rules in this Recommendation | International Standard are specified in terms of an encoding procedure. Implementations are not required to mirror the procedure specified, provided the octet string produced as the complete encoding of an abstract syntax value is identical to one of those specified in this Recommendation | International Standard for the applicable transfer syntax.

6.5 Implementations performing decoding are required to produce the abstract syntax value corresponding to any received octet string which could be produced by a sender conforming to the encoding rules identified in the transfer syntax associated with the material being decoded.

6.6 If an ASN.1 specification assigns JER encoding instructions in accordance with clauses 8 to 13 such that an ASN.1 type or component has final encoding instructions that violate the restrictions specified in clauses 14 to 19, then that ASN.1 specification is not in conformity with this Recommendation | International Standard, even if (without the encoding instructions) it would conform to all the requirements of Rec. ITU-T X.680 | ISO/IEC 8824-1.

NOTE – It is only occasionally invalid to assign an encoding instruction to a "Type", as it can be negated (removed from the set of associated encoding instructions) by a further assignment. It is the final encoding instructions that determine conformity of the specification.

7 General provisions

7.1 Use of the type notation

7.1.1 These encoding rules make specific use of the ASN.1 type notation as specified in Rec. ITU-T X.680 | ISO/IEC 8824-1, Rec. ITU-T X.681 | ISO/IEC 8824-2, Rec. ITU-T X.682 | ISO/IEC 8824-3, Rec. ITU-T X.683 | ISO/IEC 8824-4, and can only be applied to encode the values of a single ASN.1 type specified using that notation.

7.1.2 In particular, but not exclusively, they are dependent on the following information being retained in the ASN.1 type and value model underlying the use of the notation:

a) the identifiers of the components of a sequence or set type and of the alternatives of a choice type;

b) the identifiers of the enumeration items of an enumerated type;

c) whether a set or sequence type component has a default value or not;

d) the restricted range of values of a type which arise through the application of JER-visible constraints;

e) whether the type of a component is an open type.

7.2 Constraints

NOTE – The fact that some ASN.1 constraints may not be JER-visible for the purposes of encoding and decoding does not in any way affect the use of such constraints in the handling of errors detected during decoding, nor does it imply that values violating such constraints are allowed to be transmitted by a conforming sender. However, this Recommendation | International Standard makes no use of such constraints in the specification of encodings.

7.2.1 In general the constraint on a type will consist of individual constraints combined using some or all of set arithmetic, contained subtype constraints, and serial application of constraints.

The following constraints are JER-visible:

a) non-extensible size constraints on bitstring types;

b) non-extensible single value constraints on real types where the single value is either plus zero or minus zero or one of the special real values PLUS-INFINITY, MINUS-INFINITY and NOT-A-NUMBER;

c) non-extensible single value constraints and value range constraints on the base of a real type;

d) an inner type constraint that applies a non-extensible single value constraint or value range constraint to the base of a real type;

e) a contents constraint with CONTAINING but without ENCODED BY;

f) a contained subtype constraint in which the constraining type carries a JER-visible constraint.

7.2.2 All other constraints are not JER-visible. In particular, the following constraints are not JER-visible:

a) constraints that are expressed in human-readable text or in ASN.1 comment;

b) variable constraints (see Rec. ITU-T X.683 | ISO/IEC 8824-4, clauses 10.3 and 10.4);

c) user-defined constraints (see Rec. ITU-T X.682 | ISO/IEC 8824-3, 9.1);

d) table constraints (see Rec. ITU-T X.682 | ISO/IEC 8824-3);

e) component relation constraints (see Rec. ITU-T X.682 | ISO/IEC 8824-3, 10.7);

f) constraints whose evaluation is textually dependent on a table constraint or a component relation constraint (see Rec. ITU-T X.682 | ISO/IEC 8824-3);

g) extensible subtype constraints;

h) size constraints applied to a character string or octet string type;

i) single value subtype constraints applied to a character string type;

j) permitted alphabet constraints;

k) pattern constraints;

l) value and value range constraints on integer types;

m) constraints on real types except those specified in clause 7.2.1 (b) and (c);

n) constraints on the time type and on the useful and defined time types;

o) inner type constraints except those specified in clause 7.2.1 (d);

p) constraints on the useful types.

7.2.3 If a type is specified using a serial application of constraints, each of those constraints may or may not be individually JER-visible. If the last subtype constraint of the series of constraints is JER-visible and contains an extension marker, then that subtype constraint is extensible for the purposes of these encoding rules. Any other constraint is not extensible for the purposes of these encoding rules, even if it contains an extension marker.

NOTE – In a serial application of constraints, each subtype constraint removes the extensibility specified in earlier constraints of the series of constraints (see Rec. ITU-T X.680 | ISO/IEC 8824-1, 50.8).

7.2.4 If a constraint that is JER-visible is part of an **INTERSECTION** construction, then the resulting constraint is JER-visible, and consists of the **INTERSECTION** of all the JER-visible parts (with the non-JER-visible parts ignored).

7.2.5 If a constraint that is not JER-visible is part of a **UNION** construction, then the resulting constraint is not JER-visible.

7.2.6 If a constraint has an **EXCEPT** clause, the **EXCEPT** keyword and the following value set is completely ignored, whether the value set following the **EXCEPT** keyword is JER-visible or not.

7.2.7 The effective value constraint of an integer type is an integer range determined as follows, taking into account all the JER-visible constraints present in the type definition and ignoring any constraints that are not JER-visible:

a) the lower bound of the effective value constraint is the least permitted value of the integer type, if such a value exists; otherwise, the effective value constraint has no finite lower bound;

b) the upper bound of the effective value constraint is the greatest permitted value of the integer type, if such a value exists; otherwise, the effective value constraint has no finite upper bound.

NOTE – The only integer types that can have an effective value constraints with a finite lower or upper bound are the type of the components of a real type, to which a value or value range constraint is applied by using an inner type constraint. Value