INTERNATIONAL STANDARD

ISO/IEC 8824-1

Sixth edition
2021-06

Information technology — Abstract Syntax Notation One (ASN.1) —

Part 1: Specification of basic notation

Technologies de l'information — Notation de syntaxe abstraite numéro un (ASN.1) —

Partie 1: Spécification de la notation de base

Reference number
ISO/IEC 8824-1:2021(E)

Documents
Document Sharing Hub

© ISO/IEC 2021

Full document details: ISO IEC 08824-1-2021

COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2021

All rights reserved. Unless otherwise specified, or required in the context of its implementation, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

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

© ISO/IEC 2021 – All rights reserved

Documents
Document Sharing Hub

Full document details: ISO IEC 08824-1-2021

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives or www.iec.ch/members_experts/refdocs)

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents) or the IEC list of patent declarations received (see patents.iec.ch).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 6, *Telecommunications and information exchange between systems,* in collaboration with ITU-T. The identical text is published as ITU-T X.680 (02/2021).

This sixth edition cancels and replaces the fifth edition (ISO/IEC 8824-1:2015), which has been technically revised. It also incorporates ISO/IEC 8824-1:2015/Amd 1:2019, ISO/IEC 8824-1:2015/Cor 1:2017, ISO/IEC 8824-1:2015/Cor 2:2018, ISO/IEC 8824-1:2015/Cor 3:2018.

A list of all parts in the ISO/IEC 8824 series can be found on the ISO and IEC websites.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

© ISO/IEC 2021 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

Documents
Document Sharing Hub

Full document details: ISO IEC 08824-1-2021

CONTENTS

		Page
Introduction		viii
1	Scope	1
2	Normative references	1
2.1	Identical Recommendations | International Standards	1
2.2	Additional references	2
3	Definitions	2
3.1	International Object Identifier tree specification	2
3.2	Information object specification	2
3.3	Constraint specification	3
3.4	Parameterization of ASN.1 specification	3
3.5	Structure for identification of organizations	3
3.6	Universal Multiple-Octet Coded Character Set (UCS)	3
3.7	Representation of dates and times	3
3.8	Additional definitions	4
4	Abbreviations	9
5	Notation	9
5.1	General	9
5.2	Productions	10
5.3	The alternative collections	10
5.4	Non-spacing indicator	10
5.5	Example of a production	10
5.6	Layout	10
5.7	Recursion	11
5.8	References to permitted sequences of lexical items	11
5.9	References to a lexical item	11
5.10	Short-hand notations	11
5.11	Value references and the typing of values	12
6	The ASN.1 model of type extension	12
7	Extensibility requirements on encoding rules	12
8	Tags	13
9	Encoding instructions	14
10	Use of the ASN.1 notation	15
11	The ASN.1 character set	15
12	ASN.1 lexical items	16
12.1	General rules	16
12.2	Type references	17
12.3	Identifiers	17
12.4	Value references	17
12.5	Module references	17
12.6	Comments	17
12.7	Empty lexical item	18
12.8	Numbers	18
12.9	Real numbers	18
12.10	Binary strings	18
12.11	XML binary string item	18
12.12	Hexadecimal strings	18
12.13	XML hexadecimal string item	19
12.14	Character strings	19
12.15	XML character string item	20
12.16	The simple character string lexical item	22

© ISO/IEC 2021 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

12.17 Time value character strings ... 22
12.18 XML time value character string item ... 22
12.19 The property and setting names lexical item ... 22
12.20 Assignment lexical item ... 22
12.21 Range separator ... 22
12.22 Ellipsis ... 22
12.23 Left version brackets ... 23
12.24 Right version brackets ... 23
12.25 Encoding references ... 23
12.26 Integer-valued Unicode labels ... 23
12.27 Non-integer Unicode labels ... 23
12.28 XML end tag start item ... 23
12.29 XML single tag end item ... 23
12.30 XML boolean true item ... 23
12.31 XML boolean extended-true item ... 24
12.32 XML boolean false item ... 24
12.33 XML boolean extended-false item ... 24
12.34 XML real not-a-number item ... 24
12.35 XML real infinity item ... 24
12.36 XML tag names for ASN.1 types ... 25
12.37 Single character lexical items ... 26
12.38 Reserved words ... 26
13 Module definition ... 27
14 Referencing type and value definitions ... 31
15 Notation to support references to ASN.1 components ... 32
16 Assigning types and values ... 33
17 Definition of types and values ... 35
18 Notation for the boolean type ... 38
19 Notation for the integer type ... 38
20 Notation for the enumerated type ... 39
21 Notation for the real type ... 41
22 Notation for the bitstring type ... 42
23 Notation for the octetstring type ... 44
24 Notation for the null type ... 44
25 Notation for sequence types ... 45
26 Notation for sequence-of types ... 48
27 Notation for set types ... 50
28 Notation for set-of types ... 51
29 Notation for choice types ... 52
30 Notation for selection types ... 54
31 Notation for prefixed types ... 54
31.1 General ... 54
31.2 The tagged type ... 55
31.3 The encoding prefixed type ... 55
32 Notation for the object identifier type ... 56
33 Notation for the relative object identifier type ... 58
34 Notation for the OID internationalized resource identifier type ... 59
35 Notation for the relative OID internationalized resource identifier type ... 60
36 Notation for the embedded-pdv type ... 60
37 Notation for the external type ... 62

© ISO/IEC 2021 – All rights reserved
Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

38 The time type ... 63
38.1 General ... 63
38.2 Time properties and settings of time abstract values ... 63
38.3 Basic value notation and XML value notation for time abstract values with specified property settings ... 67
38.4 Useful time types ... 71
39 The character string types ... 72
40 Notation for character string types ... 73
41 Definition of restricted character string types ... 73
42 Naming characters, collections and property category sets ... 77
43 Canonical order of characters ... 81
44 Definition of unrestricted character string types ... 82
45 Notation for types defined in clauses 46 to 48 ... 83
46 Generalized time ... 83
47 Universal time ... 84
48 The object descriptor type ... 85
49 Constrained types ... 86
50 Element set specification ... 87
51 Subtype elements ... 89
51.1 General ... 89
51.2 Single value ... 90
51.3 Contained subtype ... 90
51.4 Value range ... 90
51.5 Size constraint ... 91
51.6 Type constraint ... 91
51.7 Permitted alphabet ... 91
51.8 Inner subtyping ... 92
51.9 Pattern constraint ... 93
51.10 Property settings ... 93
51.11 Duration range ... 94
51.12 Time point range ... 95
51.13 Recurrence range ... 95
52 The extension marker ... 95
53 The exception identifier ... 97
54 Encoding control sections ... 98
A.1 Definition ... 99
A.2 Metacharacters ... 99
B.1 General ... 103
B.2 The ASN.1 defined time types module ... 103
C.1 The need for the value mapping concept (tutorial introduction) ... 108
C.2 Value mappings ... 110
C.3 Identical type definitions ... 111
C.4 Specification of value mappings ... 113
C.5 Additional value mappings defined for the character string types ... 113
C.6 Specific type and value compatibility requirements ... 114
C.7 Examples ... 115
C.7.2 Example 1 ... 115
C.7.3 Example 2 ... 115
C.7.4 Example 3 ... 115
C.7.5 Example 4 ... 115
C.7.6 Example 5 ... 115
C.7.7 Example 6 ... 116

© ISO/IEC 2021 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

D.1 Values assigned in this Recommendation | International Standard ... 117
D.2 Object identifiers in the ASN.1 and encoding rules standards ... 117
F.1 General ... 120
F.2 Use of the International Object Identifier tree by the object identifier (`OBJECT IDENTIFIER`) type ... 120
F.3 Use of the International Object Identifier tree by the OID internationalized resource identifier (`OID-IRI`) type ... 120
G.1 Example of a personnel record ... 121
G.1.1 Informal description of Personnel Record ... 121
G.1.2 ASN.1 description of the record structure ... 121
G.1.3 ASN.1 description of a record value ... 122
G.2 Guidelines for use of the notation ... 122
G.2.1 Boolean ... 123
G.2.2 Integer ... 123
G.2.3 Enumerated ... 123
G.2.4 Real ... 124
G.2.5 Bit string ... 124
G.2.6 Octet string ... 126
G.2.7 UniversalString, BMPString and UTF8String ... 126
G.2.8 CHARACTER STRING ... 127
G.2.9 Null ... 128
G.2.10 Sequence and sequence-of ... 128
G.2.11 Set and set-of ... 130
G.2.12 Tagged ... 132
G.2.13 Choice ... 133
G.2.14 Selection type ... 135
G.2.16 Embedded-pdv ... 136
G.2.17 External ... 136
G.2.18 Instance-of ... 136
G.2.19 Object identifier ... 137
G.2.20 OID internationalized resource identifier ... 137
G.2.21 Relative object identifier ... 137
G.3 Value notation and property settings (`TIME` type and useful time types) ... 137
G.3.1 Date ... 137
G.3.2 Time of day ... 138
G.3.3 Date and time of day ... 138
G.3.4 Time interval ... 139
G.3.5 Recurring interval ... 140
G.4 Identifying abstract syntaxes ... 140
G.5 Subtypes ... 141
H.1 Character string support in ASN.1 ... 145
H.2 The UniversalString, UTF8String and BMPString types ... 145
H.3 On ISO/IEC 10646 conformance requirements ... 146
H.4 Recommendations for ASN.1 users on ISO/IEC 10646 conformance ... 146
H.5 Adopted subsets as parameters of the abstract syntax ... 147
H.6 The CHARACTER STRING type ... 147
I.1 Overview ... 148
I.2 Meaning of version numbers ... 149
I.3 Requirements on encoding rules ... 150
I.4 Combination of (possibly extensible) constraints ... 150
I.4.1 Model ... 150
I.4.2 Serial application of constraints ... 150
I.4.3 Use of set arithmetic ... 151
I.4.4 Use of the Contained Subtype notation ... 152
J.1 The collections of ASN.1 types for times and dates ... 153
J.2 ISO 8601 key concepts ... 153
J.3 Abstract values of the `TIME` type ... 154
J.4 Time properties of the time abstract values ... 155
J.5 Value notation ... 155

© ISO/IEC 2021 – All rights reserved
Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

J.6 Use of the ASN.1 subtype notation 156
J.7 The property settings subtype notation 156
K.1 General 158
K.2 Analyzing the full string 158
K.3 Analysis of a string containing an interval 159
K.4 Analysis of a string containing a date 159
K.5 Analysis of a string containing a year 160
K.6 Analysis of a string containing a century 160
K.7 Analysis of a string containing a time 160
K.8 Analysis of a string containing a simple time 161

© ISO/IEC 2021 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

Introduction

This Recommendation | International Standard presents a standard notation for the definition of data types and values. A *data type* (or *type* for short) is a category of information (for example, numeric, textual, still image or video information). A *data value* (or *value* for short) is an instance of such a type. This Recommendation | International Standard defines several basic types and their corresponding values, and rules for combining them into more complex types and values.

In some protocol architectures, each message is specified as the binary value of a sequence of octets. However, standards-writers need to define quite complex data types to carry their messages, without concern for their binary representation. In order to specify these data types, they require a notation that does not necessarily determine the representation of each value. ASN.1 is such a notation. This notation is supplemented by the specification of one or more algorithms called *encoding rules* that determine the value of the octets that carry the application semantics (called the *transfer syntax*). Rec. ITU-T X.690 | ISO/IEC 8825-1, Rec. ITU-T X.691 | ISO/IEC 8825-2 and Rec. ITU-T X.693 | ISO/IEC 8825-4 specify three families of standardized encoding rules, called *Basic Encoding Rules (BER)*, *Packed Encoding Rules (PER)*, and *XML Encoding Rules (XER)*.

Some users wish to redefine their legacy protocols using ASN.1, but cannot use standardized encoding rules because they need to retain their existing binary representations. Other users wish to have more complete control over the exact layout of the bits on the wire (the transfer syntax). These requirements are addressed by Rec. ITU-T X.692 | ISO/IEC 8825-3 which specifies an *Encoding Control Notation (ECN)* for ASN.1. ECN enables designers to formally specify the abstract syntax of a protocol using ASN.1, but to then (if they so wish) take complete or partial control of the bits on the wire by writing an accompanying ECN specification (which may reference standardized Encoding Rules for some parts of the encoding).

A very general technique for defining a complicated type at the abstract level is to define a small number of *simple types* by defining all possible values of the simple types, then combining these simple types in various ways. Some of the ways of defining new types are as follows:

a) given an (ordered) list of existing types, a value can be formed as an (ordered) sequence of values, one from each of the existing types; the collection of all possible values obtained in this way is a new type (if the existing types in the list are all distinct, this mechanism can be extended to allow omission of some values from the list);

b) given an unordered set of (distinct) existing types, a value can be formed as an (unordered) set of values, one from each of the existing types; the collection of all possible unordered sets of values obtained in this way is a new type (the mechanism can again be extended to allow omission of some values);

c) given a single existing type, a value can be formed as an (ordered) list or (unordered) set of zero, one or more values of the existing type; the collection of all possible lists or sets of values obtained in this way is a new type;

d) given a list of (distinct) types, a value can be chosen from any one of them; the set of all possible values obtained in this way is a new type;

e) given a type, a new type can be formed as a subset of it by using some structure or order relationship among the values.

An important aspect of combining types in this way is that encoding rules should recognize the combining constructs, providing unambiguous encodings of the collection of values of the basic types. Thus, every basic type defined using the notation specified in this Recommendation | International Standard is assigned a *tag* to aid in the unambiguous encoding of values.

Tags are mainly intended for machine use, and are not essential for the human notation defined in this Recommendation | International Standard. Where, however, it is necessary to require that certain types be distinct, this is expressed by requiring that they have distinct tags. The allocation of tags is therefore an important part of the use of this notation, but (since 1994) it is possible to specify the automatic allocation of tags.

NOTE 1 – Within this Recommendation | International Standard, tag values are assigned to all simple types and construction mechanisms. The restrictions placed on the use of the notation ensure that tags can be used in transfer for unambiguous identification of values.

It is also possible to assign encoding instructions to a type in order to affect the encoding of that type. This can be done either by a type prefix placed before a type definition or use of a type reference, or by an encoding control section placed at the end of an ASN.1 module. The generic syntax of type prefixes and encoding control sections is specified in this Recommendation | International Standard, and includes an encoding reference to identify the encoding rules that are modified by the encoding instruction. The semantics and detailed syntax of encoding instructions are specified in the encoding rules Recommendation | International Standard identified by the encoding reference.

© ISO/IEC 2021 – All rights reserved
Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

An ASN.1 specification will initially be produced with a set of fully defined ASN.1 types. At a later stage, however, it may be necessary to change those types (usually by the addition of extra components in a sequence or set type). If this is to be possible in such a way that implementations using the old type definitions can interwork with implementations using the new type definitions in a defined way, encoding rules need to provide appropriate support. The ASN.1 notation supports the inclusion of an *extension marker* on a number of types. This signals to encoding rules the intention of the designer that this type is one of a series of related types (i.e., versions of the same initial type) called an *extension series*, and that the encoding rules are required to enable information transfer between implementations using different types that are related by being part of the same extension series.

Clauses 11 to 33 (inclusive) define the simple types supported by ASN.1, and specify the notation to be used for referencing simple types and for defining new types using them. Clauses 11 to 33 also specify notations to be used for specifying values of types defined using ASN.1. Two value notations are provided. The first is called the basic ASN.1 value notation, and has been part of the ASN.1 notation since its first introduction. The second is called the XML ASN.1 Value Notation, and provides a value notation using Extensible Markup Language (XML).

NOTE 2 – The XML Value Notation provides a means of representing ASN.1 values using XML. Thus, an ASN.1 type definition also specifies the structure and content of an XML element. This makes ASN.1 a simple schema language for XML.

Clauses 36 to 37 (inclusive) define the types supported by ASN.1 for carrying within them the complete encoding of ASN.1 types.

Clause 38 and Annex B define the types that provide support for ISO 8601.

Clauses 39 to 44 (inclusive) define the character string types.

Clauses 45 to 48 (inclusive) define certain types which are considered to be of general utility, but which require no additional encoding rules.

Clauses 49 to 51 (inclusive) define a notation which enables subtypes to be defined from the values of a parent type.

Clause 52 defines a notation which allows ASN.1 types specified in a "version 1" specification to be identified as likely to be extended in "version 2", and for additions made in subsequent versions to be separately listed and identified with their version number.

Clause 53 defines a notation which allows ASN.1 type definitions to contain an indication of the intended error handling if encodings are received for values which lie outside those specified in the current standardized definition.

Annex A forms an integral part of this Recommendation | International Standard, and specifies ASN.1 regular expressions.

Annex B forms an integral part of this Recommendation | International Standard, and defines an ASN.1 module containing the definition of a set of time types providing the full functionality of ISO 8601. These types can be imported from this ASN.1 module by an application designer if the useful time types specified in clause 38 are not adequate for the application.

Annex C forms an integral part of this Recommendation | International Standard, and specifies rules for type and value compatibility.

Annex D forms an integral part of this Recommendation | International Standard, and records object identifier and object descriptor values assigned in the ASN.1 series of Recommendations | International Standards.

Annex E forms an integral part of this Recommendation | International Standard and specifies the currently defined encoding references and the Recommendation | International Standard that defines the semantics and detailed syntax of encoding instructions with those encoding references.

Annex F does not form an integral part of this Recommendation | International Standard, and references the specification of the top-level arcs of the International Object Identifier tree and the use of that tree to form an OID internationalized resource identifier which can be used as an IRI or URI registered as the "oid" scheme with IANA.

Annex G does not form an integral part of this Recommendation | International Standard, and provides examples and hints on the use of the ASN.1 notation.

Annex H does not form an integral part of this Recommendation | International Standard, and provides a tutorial on ASN.1 character strings.

Annex I does not form an integral part of this Recommendation | International Standard, and provides a tutorial on the ASN.1 model of type extension.

Annex J does not form an integral part of this Recommendation | International Standard and provides a tutorial introduction to ISO 8601 and to the **`TIME`** type. It is recommended that this be read before the normative text.

© ISO/IEC 2021 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

Annex K does not form an integral part of this Recommendation | International Standard and provides information on how to identify the time properties of an abstract value from an instance of value notation.

Annex L does not form an integral part of this Recommendation | International Standard, and provides a summary of ASN.1 using the notation of clause 5.

© ISO/IEC 2021 – All rights reserved
Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

INTERNATIONAL STANDARD
ITU-T RECOMMENDATION

Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation

1 Scope

This Recommendation | International Standard provides a standard notation called Abstract Syntax Notation One (ASN.1) that is used for the definition of data types, values, and constraints on data types.

This Recommendation | International Standard:

- defines a number of simple types, with their tags, and specifies a notation for referencing these types and for specifying values of these types;
- defines mechanisms for constructing new types from more basic types, and specifies a notation for defining such types and assigning them tags, and for specifying values of these types;
- defines character sets (by reference to other Recommendations and/or International Standards) for use within ASN.1.

The ASN.1 notation can be applied whenever it is necessary to define the abstract syntax of information.

The ASN.1 notation is referenced by other standards which define encoding rules for the 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.

NOTE – This Recommendation | International Standard is based on ISO/IEC 10646:2003 and the Unicode standard version 3.2.0:2002. It cannot be applied using later versions of these two standards.

2.1 Identical Recommendations | International Standards

- Recommendation ITU-T X.660 (2011) | ISO/IEC 9834-1:2012, *Information technology – Open Systems Interconnection – Procedures for the operation of OSI Registration Authorities: General procedures and top arcs of the ASN.1 International Object Identifier tree*.
- Recommendation ITU-T X.681 (2021) | ISO/IEC 8824-2:2021, *Information technology – Abstract Syntax Notation One (ASN.1): Information object specification*.
- Recommendation ITU-T X.682 (2021) | ISO/IEC 8824-3:2021, *Information technology – Abstract Syntax Notation One (ASN.1): Constraint specification*.
- Recommendation ITU-T X.683 (2021) | ISO/IEC 8824-4:2021, *Information technology – Abstract Syntax Notation One (ASN.1): Parameterization of ASN.1 specifications*.
- Recommendation ITU-T X.690 (2021) | ISO/IEC 8825-1:2021, *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 (2021) | ISO/IEC 8825-2:2021, *Information technology – ASN.1 encoding rules: Specification of Packed Encoding Rules (PER)*.
- Recommendation ITU-T X.692 (2021) | ISO/IEC 8825-3:2021, *Information technology – ASN.1 encoding rules: Specification of Encoding Control Notation (ECN)*.
- Recommendation ITU-T X.693 (2021) | ISO/IEC 8825-4:2021, *Information technology – ASN.1 encoding rules: XML Encoding Rules (XER)*.

© ISO/IEC 2021 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

– Recommendation ITU-T X.695 (2021) | ISO/IEC 8825-6:2021, *Information technology – ASN.1 encoding rules: Registration and application of PER encoding instructions*.

NOTE – The references above shall be interpreted as references to the identified Recommendations | International Standards together with all their published amendments and technical corrigenda.

2.2 Additional references

– Recommendation ITU-R TF.460-6 (2002), *Standard-frequency and time-signal emissions.*

– CCITT Recommendation T.100 (1988), *International information exchange for interactive Videotex.*

– Recommendation ITU-T T.101 (1994), *International interworking for Videotex services.*

– ISO *International Register of Coded Character Sets to be used with Escape Sequences*.

– ISO/IEC 646:1991, *Information technology – ISO 7-bit coded character set for information interchange*.

– ISO/IEC 2022:1994, *Information technology – Character code structure and extension techniques*.

– ISO/IEC 6523:1998, *Data interchange – Structures for the identification of organizations and organization parts*.

– ISO/IEC 7350:1991, *Information technology – Registration of repertoires of graphic characters from ISO/IEC 10367*.

– ISO 8601:2019, *Date and time – Representations for information interchange*.

– ISO/IEC 10646:2003, *Information technology – Universal Multiple-Octet Coded Character Set (UCS)*.

– The Unicode Standard, Version 3.2.0:2002. The Unicode Consortium. (Reading, MA, Addison-Wesley)

NOTE 1 – The above reference is included because it provides names for control characters and specifies categories of characters.

– W3C XML 1.0:2008, *Extensible Markup Language (XML) 1.0 (Fifth Edition), W3C Recommendation, Copyright ©2008 W3C, (MIT, ERCIM, Keio)*, http://www.w3.org/TR/2008/REC-xml-20081126/.

NOTE – The reference to a document within this Recommendation | International Standard does not give it, as a stand-alone document, the status of a Recommendation or International Standard.

3 Definitions

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

3.1 International Object Identifier tree specification

This Recommendation | International Standard uses the following terms defined in Rec. ITU-T X.660 | ISO/IEC 9834-1:

a) integer-valued Unicode label;

b) international object identifier tree;

c) OID internationalized resource identifier;

d) long arc;

e) object identifier;

f) primary integer value;

g) secondary identifier;

h) Unicode label.

3.2 Information object specification

This Recommendation | International Standard uses the following terms defined in Rec. ITU-T X.681 | ISO/IEC 8824-2:

a) information object;

b) information object class;

c) information object set;

d) instance-of type;

e) object class field type.

© ISO/IEC 2021 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

3.3 Constraint specification

This Recommendation | International Standard uses 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 uses the following terms defined in Rec. ITU-T X.683 | ISO/IEC 8824-4:

a) parameterized type;

b) parameterized value.

3.5 Structure for identification of organizations

This Recommendation | International Standard uses the following terms defined in ISO/IEC 6523:

a) issuing organization;
b) organization code;
c) International Code Designator.

3.6 Universal Multiple-Octet Coded Character Set (UCS)

This Recommendation | International Standard uses the following terms defined in ISO/IEC 10646:

a) Basic Multilingual Plane (BMP);
b) cell;
c) combining character;
d) graphic symbol;
e) group;
f) limited subset;
g) plane;
h) row;
i) selected subset.

3.7 Representation of dates and times

This Recommendation | International Standard uses the following terms defined in ISO 8601:

a) basic format;

b) calendar date;

c) common year;

d) duration;

e) extended format;

f) Gregorian calendar;

g) instant;

h) leap second;

i) leap year;

j) local time of day;

k) ordinal date;

l) recurring time interval

m) time axis;

n) time interval;

o) time;

p) time-scale;

q) UTC;

© ISO/IEC 2021 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

r) week date.

3.8 Additional definitions

3.8.1 abstract character: An abstract value which is used for the organization, control or representation of textual data.

NOTE – Annex H provides a more complete description of the term abstract character.

3.8.2 abstract value: A value whose definition is based only on the type used to carry some semantics, independently of how it is represented in any encoding.

NOTE – Examples of abstract values are the values of the integer type, the boolean type, a character string type, or of a type which is a sequence (or a choice) of an integer and a boolean.

3.8.3 additional time type: A type defined as a subtype of the time type (see 3.8.83) by applying the property setting subtype notation to the time type or to a useful or defined time type.

3.8.4 ASN.1 character set: The set of characters, specified in clause 11, used in the ASN.1 notation.

3.8.5 ASN.1 specification: A collection of one or more ASN.1 modules.

3.8.6 associated type: A type which is used only for defining the value and subtype notation for a type.

NOTE – Associated types are defined in this Recommendation | International Standard when it is necessary to make it clear that there may be a significant difference between how the type is defined in ASN.1 and how it is encoded. Associated types do not appear in user specifications.

3.8.7 bitstring type: A simple type whose distinguished values are an ordered sequence of zero, one or more bits.

NOTE – Where there is a need to carry embedded encodings of an abstract value, the use of a bitstring (or an octetstring) type without a contents constraint (see Rec. ITU-T X.682 | ISO/IEC 8824-3, clause 11) is deprecated. Otherwise, the use of the embedded-pdv type (see clause 36) provides a more flexible mechanism, allowing the announcement of the abstract syntax and of the encoding of the abstract value that is embedded.

3.8.8 boolean type: A simple type with two distinguished values.

3.8.9 character property: The set of information associated with a cell in a table defining a character repertoire.

NOTE – The information will normally include some or all of the following items:

a) a graphic symbol;
b) a character name;
c) the definition of functions associated with the character when used in particular environments;
d) whether it represents a digit;
e) an associated character differing only in (upper/lower) case.

3.8.10 character abstract syntax: Any abstract syntax whose values are specified as the set of character strings of zero, one or more characters from some specified collection of characters.

3.8.11 character repertoire: The characters in a character set without any implication on how such characters are encoded.

3.8.12 character string types: Simple types whose values are strings of characters from some defined character set.

3.8.13 character transfer syntax: Any transfer syntax for a character abstract syntax.

3.8.14 choice types: Types defined by referencing a list of distinct types; each value of the choice type is derived from the value of one of the component types.

3.8.15 component type: One of the types referenced when defining a **`CHOICE`**, **`SET`**, **`SEQUENCE`**, **`SET OF`**, or **`SEQUENCE OF`**.

3.8.16 constraint: A notation which can be used in association with a type, to define a subtype of that type.

3.8.17 contents constraint: A constraint on a bit string or octet string type that specifies either that the contents are to be an encoding of a specified ASN.1 type, or that specified procedures are to be used to produce and process the contents.

3.8.18 control characters: Characters appearing in some character repertoires that have been given a name (and perhaps a defined function in relation to certain environments) but which have not been assigned a graphic symbol, and which are not spacing characters.

NOTE – HORIZONTAL TABULATION (9) and LINE FEED (10) are examples of control characters that have been assigned a formatting function in a printing environment. DATA LINK ESCAPE (16) is an example of a control character that has been assigned a function in a communication environment.

© ISO/IEC 2021 – All rights reserved
Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

3.8.19 Coordinated Universal Time (UTC): The time scale maintained by the Bureau International de l'Heure (International Time Bureau) that forms the basis of a coordinated dissemination of standard frequencies and time signals.

NOTE 1 – The source of this definition is Rec. ITU-R TF.460-5. ITU-R has also defined the acronym for Coordinated Universal Time as UTC.

NOTE 2 – UTC and Greenwich Mean Time (GMT) are two alternative time standards which for most practical purposes determine the same time.

3.8.20 default encoding reference (for a module): An encoding reference that is specified in the module header and is assumed in all type prefixes which do not contain an encoding reference.

NOTE – If a default encoding reference is not specified in the module header, then all type prefixes which do not contain an encoding reference are assigning tags.

3.8.21 defined time type: A type defined in Annex B as a subtype of the time type (see 3.8.83) that is intended for importation by application designers when needed for their application.

3.8.22 element: A value of a governing type or an information object of a governing information object class, distinguable from all other values of the same type or information objects of the same class, respectively.

3.8.23 element set: A set of elements, all of which are values of a governing type, or information objects of a governing class.

NOTE – Governing class is defined in Rec. ITU-T X.681 | ISO/IEC 8824-2, 3.4.7.

3.8.24 embedded-pdv type: A type whose set of values is formally the union of the sets of values in all possible abstract syntaxes. This type can be used in an ASN.1 specification that wishes to carry in its protocol an abstract value whose type may be defined externally to that ASN.1 specification. It carries an identification of the abstract syntax (the type) of the abstract value being carried, as well as an identification of the encoding rules used to encode that abstract value.

3.8.25 encoding: The bit-pattern resulting from the application of a set of encoding rules to an abstract value.

3.8.26 encoding control section: Part of an ASN.1 module that enables encoding instructions to be assigned to types defined or used within that ASN.1 module.

3.8.27 encoding instruction: Information which can be associated with a type using a type prefix or an encoding control section, and which affects the encoding of that type by one or more ASN.1 encoding rules.

NOTE – An encoding instruction does not affect the abstract values of a type, and is not expected to be visible to an application.

3.8.28 encoding reference: A name (see Annex E) that identifies which encoding rules are affected by an encoding instruction in a type prefix or an encoding control section.

NOTE – The encoding reference **TAG** can be used to specify that a type prefix is assigning a tag rather than an encoding instruction (see 31.2).

3.8.29 (ASN.1) encoding rules: Rules which specify the representation during transfer of the values of ASN.1 types. Encoding rules also enable the values to be recovered from the representation, given knowledge of the type.

NOTE – For the purpose of specifying encoding rules, the various referenced type (and value) notations, which can provide alternative notations for built-in types (and values), are not relevant.

3.8.30 enumerated types: Simple types whose values are given distinct identifiers as part of the type notation.

3.8.31 extension addition: One of the added notations in an extension series. For set, sequence and choice types, each extension addition is the addition of either a single extension addition group or a single component type. For enumerated types it is the addition of a single further enumeration. For a constraint it is the addition of (only) one subtype element.

NOTE – Extension additions are both textually ordered (following the extension marker) and logically ordered (having increasing enumeration values, and, in the case of **CHOICE** alternatives, increasing tags).

3.8.32 extension addition group: One or more components of a set, sequence or choice type grouped within version brackets. An extension addition group is used to clearly identify the components of a set, sequence or choice type that were added in a particular version of an ASN.1 module, and can identify that version with a simple integer.

3.8.33 extension addition type: A type contained within an extension addition group or a single component type that is itself an extension addition (in such a case it is not contained within an extension addition group).

3.8.34 extensible constraint: A subtype constraint with an extension marker at the outer level, or that is extensible through the use of set arithmetic with extensible sets of values.

3.8.35 extension insertion point (or insertion point): The location within a type definition where extension additions are inserted. This location is the end of the type notation of the immediately preceding type in the extension series if there is a single ellipsis in the type definition, or immediately before the second ellipsis if there is an extension marker pair in the definition of the type.

© ISO/IEC 2021 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

NOTE – There can be at most one insertion point within the components of any choice, sequence, or set type.

3.8.36 extension marker: A syntactic flag (an ellipsis) that is included in all types that form part of an extension series.

3.8.37 extension marker pair: A pair of extension markers between which extension additions are inserted.

3.8.38 extension-related: Two types that have the same extension root, where one was created by adding zero or more extension additions to the other.

3.8.39 extension root: An extensible type that is the first type in an extension series. It carries either the extension marker with no additional notation other than comments and white-space between the extension marker and the matching "}" or ")", or an extension marker pair with no additional notation other than a single comma, comments and white-space between the extension markers.

NOTE – Only an extension root can be the first type in an extension series.

3.8.40 extension series: A series of ASN.1 types which can be ordered in such a way that each successive type in the series is formed by the addition of text at the extension insertion point.

3.8.41 extensible type: A type with an extension marker, or to which an extensible constraint has been applied.

NOTE – An extension marker can be textually present or can be inserted by an EXTENSIBILITY-IMPLIED (see 13.4).

3.8.42 external reference: A type reference, value reference, information object class reference, information object reference, or information object set reference (which may be parameterized), that is defined in some other module than the one in which it is being referenced, and which is being referred to by prefixing the module name to the referenced item.

EXAMPLE – **`ModuleName.TypeReference`**

3.8.43 external type: A type which is a part of an ASN.1 specification that carries a value whose type may be defined externally to that ASN.1 specification. It also carries an identification of the type of the value being carried.

3.8.44 false: One of the distinguished values of the boolean type (see also "true").

3.8.45 governing (type); governor: A type definition or reference which affects the interpretation of a part of the ASN.1 syntax, requiring that part of the ASN.1 syntax to reference values in the governing type.

3.8.46 identical type definitions: Two instances of the ASN.1 "Type" production (see clause 17) are defined as identical type definitions if, after performing the transformations specified in Annex C, they are identical ordered lists of identical lexical items (see clause 12).

3.8.47 OID internationalized resource identifier type: The set of all OID internationalized resource identifiers.

NOTE 1 – This is a simple type whose values are a sequence of Unicode labels that identify a series of arcs leading from the root to a node of the International Object Identifier tree, as specified by the Rec. ITU-T X.660 | ISO/IEC 9834-series.

NOTE 2 – The rules of Rec. ITU-T X.660 | ISO/IEC 9834-1 permit a wide range of authorities to independently associate Unicode labels with an arc of the tree.

3.8.48 integer type: A simple type with distinguished values which are the positive and negative whole numbers, including zero (as a single value).

NOTE – When particular encoding rules limit the range of an integer, such limitations are chosen so as not to affect any user of ASN.1.

3.8.49 lexical item: A named sequence of characters from the ASN.1 character set, specified in clause 12, which is used in forming the ASN.1 notation.

3.8.50 module: One or more instances of the use of the ASN.1 notation for type, value, value set, information object class, information object, and information object set (as well as the parameterized variant of those), encapsulated using the ASN.1 module notation (see clause 13).

NOTE – The terms information object class (etc.) are specified in Rec. ITU-T X.681 | ISO/IEC 8824-2, and parameterization is specified in Rec. ITU-T X.683 | ISO/IEC 8824-4.

3.8.51 null type: A simple type consisting of a single value, also called null.

3.8.52 object: A well-defined piece of information, definition, or specification which requires a name in order to identify its use in an instance of communication.

NOTE – Such an object may be an information object as defined in Rec. ITU-T X.681 | ISO/IEC 8824-2.

3.8.53 object descriptor type: A type whose distinguished values are human-readable text providing a brief description of an object (see 3.8.52).

NOTE – An object descriptor value is usually associated with a single object. Only an object identifier value unambiguously identifies an object.

© ISO/IEC 2021 – All rights reserved
Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

3.8.54 object identifier type: A simple type whose values are a sequence of primary integer values that identify a series of arcs leading from the root to a node of the International Object Identifier tree, as specified by the Rec. ITU-T X.660 | ISO/IEC 9834 series.

NOTE 1 – The rules of Rec. ITU-T X.660 | ISO/IEC 9834-1 permit a wide range of authorities to independently associate a primary integer value with an arc of the tree.

NOTE 2 – In the value notation for the object identifier type (and in XML encodings of that type) it is possible to include secondary identifiers for arcs.

3.8.55 octetstring type: A simple type whose distinguished values are an ordered sequence of zero, one or more octets, each octet being an ordered sequence of eight bits.

3.8.56 open systems interconnection: An architecture for computer communication which provides a number of terms which are used in this Recommendation | International Standard preceded by the abbreviation "OSI".

NOTE – The meaning of such terms can be obtained from the Rec. ITU-T X.200 series and equivalent ISO/IEC Standards if needed. The terms are only applicable if ASN.1 is used in an OSI environment.

3.8.57 open type notation: An ASN.1 notation used to denote a set of values from more than one ASN.1 type.

NOTE 1 – The term "open type" is used synonymously with "open type notation" in the body of this Recommendation | International Standard.

NOTE 2 – All ASN.1 encoding rules provide unambiguous encodings for the values of a single ASN.1 type. They do not necessarily provide unambiguous encodings for "open type notation", which carries values from ASN.1 types that are not normally determined at specification time. Knowledge of the type of the value being encoded in the "open type notation" is needed before the abstract value for that field can be unambiguously determined.

NOTE 3 – The only notation in this Recommendation | International Standard which is an open type notation is the "ObjectClassFieldType" specified in Rec. ITU-T X.681 | ISO/IEC 8824-2, clause 14, where the "FieldName" denotes either a type field or a variable-type value field.

3.8.58 parent type (of a subtype): The type that is being constrained when defining a subtype, and which governs the subtype notation.

NOTE – The parent type may itself be a subtype of some other type.

3.8.59 production: A part of the formal notation (also called grammar or Backus-Naur Form, BNF) used to specify ASN.1.

3.8.60 real type: A simple type whose distinguished values (specified in clause 21) include the set of real numbers (numerical real numbers) together with special values such as **`NOT-A-NUMBER`**.

3.8.61 recursive definition (of a type): A set of ASN.1 definitions which cannot be reordered so that all types used in a construction are defined before the definition of the construction.

NOTE – Recursive definitions are allowed in ASN.1: the user of the notation has the responsibility for ensuring that those values (of the resulting types) which are used have a finite representation and that the value set associated with the type contains at least one value.

3.8.62 relative OID internationalized resource identifier type: A value which identifies an object by its position relative to some known OID internationalized resource identifier.

3.8.63 relative object identifier: A value which identifies an object by its position relative to some known object identifier.

3.8.64 relative object identifier type: A simple type whose values are the set of all possible relative object identifiers.

3.8.65 restricted character string type: A character string type whose characters are taken from a fixed character repertoire identified in the type specification.

3.8.66 selection types: Types defined by reference to a component type of a choice type, and whose values are precisely the values of that component type.

3.8.67 sequence types: Types defined by referencing a fixed, ordered list of types (some of which may be declared to be optional); each value of the sequence type is an ordered list of values, one from each component type.

NOTE – Where a component type is declared to be optional, a value of the sequence type need not contain a value of that component type.

3.8.68 sequence-of types: Types defined by referencing a single component type; each value in the sequence-of type is an ordered list of zero, one or more values of the component type.

3.8.69 serial application (of constraints): The application of a constraint to a parent type which is already constrained.

© ISO/IEC 2021 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

3.8.70 set arithmetic: The formation of new sets of values or information objects using the operations of union, intersection and set difference (use of **EXCEPT**) as specified in 50.2.

NOTE – The result of serial application of constraints is not covered by the term "set arithmetic".

3.8.71 setting (of a time property): One of a number of values that can be associated with a given time property (see 3.8.82 and the note in J.4.2).

NOTE – Any time property that applies to a particular time abstract value has only a single setting (see Table 6).

3.8.72 set types: Types defined by referencing a fixed, unordered, list of types (some of which may be declared to be optional); each value in the set type is an unordered list of values, one from each component type.

NOTE – Where a component type is declared to be optional, a value of the set type need not contain a value of that component type.

3.8.73 set-of types: Types defined by referencing a single component type; each value in the set-of type is an unordered list of zero, one or more values of the component type.

3.8.74 simple types: Types defined by directly specifying the set of their values.

3.8.75 spacing character: A character in a character repertoire which is intended for inclusion with graphic characters in the printing of a character string but which is represented in the physical rendition by empty space; it is not normally considered to be a control character (see 3.8.18).

NOTE – There may be a single spacing character in the character repertoire, or there may be multiple spacing characters with varying widths.

3.8.76 subtype (of a parent type): A type whose values are a subset (or the complete set) of the values of some other type (the parent type).

3.8.77 tag: Additional information, separate from the abstract values of the type, which is associated with every ASN.1 type and which can be changed or augmented by a type prefix.

NOTE – Tag information is used in some encoding rules to ensure that encodings are not ambiguous. Tag information differs from encoding instructions because tag information is associated with all ASN.1 types, even if they do not have a type prefix.

3.8.78 tagged types: A type defined by referencing a single existing type and a tag; the new type is isomorphic to the existing type, but is distinct from it.

3.8.79 tagging: Assigning a new tag to a type, replacing or adding to the existing (possibly the default) tag.

3.8.80 time abstract value: An abstract value of the time type.

3.8.81 time component: Part of the definition of a time abstract value that specifies a part of that abstract value.

NOTE – Examples of time components are a date component (that would have a year component), a time-of-day component, or a time difference component.

3.8.82 time property (of a time abstract value): One of a number of terms used to describe a time abstract value (see 3.8.80).

NOTE – The time properties that can be used to describe a time abstract value often depend on the setting of some other time property of that abstract value. The time properties are listed in Table 6, column 1.

3.8.83 time type: The **TIME** type that supports all the abstract values implicitly defined by ISO 8601.

3.8.84 transfer syntax: The set of bit strings used to exchange the abstract values in an abstract syntax, usually obtained by application of encoding rules to an abstract syntax.

NOTE – The term "transfer syntax" is synonymous with "encoding".

3.8.85 true: One of the distinguished values of the boolean type (see also "false").

3.8.86 type: A named set of values.

3.8.87 type prefix: Part of the ASN.1 notation that can be used to assign an encoding instruction or a tag to a type.

3.8.88 type reference name: A name associated uniquely with a type within some context.

NOTE – Reference names are assigned to the types defined in this Recommendation | International Standard; these are universally available within ASN.1. Other reference names are defined in other Recommendations | International Standards, and are applicable only in the context of that Recommendation | International Standard.

3.8.89 unrestricted character string type: A type whose abstract values are values from a character abstract syntax, together with an identification of the character abstract syntax and of the character transfer syntax to be used in its encoding.

3.8.90 useful time type: A built-in type defined as a subtype of the time type (see 3.8.83) that is intended for direct use by application designers.

© ISO/IEC 2021 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

3.8.91 user (of ASN.1): The individual or organization that defines the abstract syntax of a particular piece of information using ASN.1.

3.8.92 value mapping: A 1-1 relationship between values in two types that enables a reference to one of those values to be used as a reference to the other value. This can, for example, be used in specifying subtypes and default values (see Annex C).

3.8.93 value reference name: A name associated uniquely with a value within some context.

3.8.94 value set: A collection of values of a type. Semantically equivalent to a subtype.

3.8.95 version brackets: A pair of adjacent left and right brackets ("`[[`" or "`]]`") used to delineate the start and end of an extension addition group. The pair of left brackets can optionally be followed by a number giving a version number for the extension addition group.

3.8.96 version number: A number which can be associated with a version bracket (see I.1.8).

NOTE – A version number cannot be added to an extension addition which is not part of an extension addition group, nor to extension additions to any type other than choice, sequence, or set.

3.8.97 white-space: Any formatting action that yields a space on a printed page, such as spaces or tabs.

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
BMP	Basic Multilingual Plane
DCC	Data Country Code
DNIC	Data Network Identification Code
ECN	Encoding Control Notation of ASN.1
ICD	International Code Designator
IRI	Internationalized Resource Identifier
OID	Object Identifier
OSI	Open Systems Interconnection
PER	Packed Encoding Rules of ASN.1
ROA	Recognized Operating Agency
UCS	Universal Multiple-Octet Coded Character Set
URI	Universal Resource Identifier
UTC	Coordinated Universal Time
XML	Extensible Markup Language

5 Notation

5.1 General

5.1.1 The ASN.1 notation consists of a sequence of characters from the ASN.1 character set specified in clause 11.

5.1.2 Each use of the ASN.1 notation contains characters from the ASN.1 character set grouped into lexical items. Clause 12 specifies all the sequences of characters forming lexical items, and names each item.

5.1.3 The ASN.1 notation is specified in clause 13 (and following clauses) by specifying and naming those sequences of lexical items which form valid instances of the ASN.1 notation, and by specifying the ASN.1 semantics of each sequence.

5.1.4 In order to specify the permitted sequences of lexical items, this Recommendation | International Standard uses a formal notation defined in the following subclauses.

© ISO/IEC 2021 – All rights reserved

Documents Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

5.2 Productions

5.2.1 All lexical items are named (see clause 12), and permitted sequences of lexical items are named.

5.2.2 A new (more complex) permitted sequence of lexical items is defined by means of a production. This uses the names of lexical items and of permitted sequences of lexical items and forms a new named permitted sequence of lexical items.

5.2.3 Each production consists of the following parts, on one or several lines, in order:

a) a name for the new permitted sequence of lexical items;

b) the characters

::=

c) one or more alternative sequences of lexical items, as defined in 5.3, separated by the character

|

5.2.4 A sequence of lexical items is present in the new permitted sequence of lexical items if it is present in one or more of the alternatives. The new permitted sequence of lexical items is referenced in this Recommendation | International Standard by the name in 5.2.3 a) above.

NOTE – If the same sequence of lexical items appears in more than one alternative, any semantic ambiguity in the resulting notation is resolved by associated text.

5.3 The alternative collections

5.3.1 Each alternative in a production (see 5.2.3.c) is specified by a list of names. Each name is either the name of a lexical item, or is the name of a permitted sequence of lexical items defined and named by some other production.

5.3.2 The permitted sequence of lexical items defined by each alternative consists of all sequences obtained by taking any one of the sequences (or the lexical item) associated with the first name, in combination with (and followed by) any one of the sequences (or lexical item) associated with the second name, in combination with (and followed by) any one of the sequences (or lexical item) associated with the third name, and so on up to and including the last name (or lexical item) in the alternative.

5.4 Non-spacing indicator

If the non-spacing indicator "**&**" (AMPERSAND) is inserted between these items in production sequences, then the lexical item that precedes it and the lexical item that follows it shall not be separated by white-space.

NOTE – This indicator is only used in productions that describe the XML value notation. For example, it is used to specify that the lexical item "<" is to be immediately followed by an XML tag name.

5.5 Example of a production

5.5.1 The production:

```
ExampleProduction        ::=
    bstring
|   hstring
|   "{" IdentifierList "}"
```

associates the name "ExampleProduction" with the following sequences of lexical items:

a) any "bstring" (a lexical item); or

b) any "hstring" (a lexical item); or

c) any sequence of lexical items associated with "IdentifierList", preceded by a "**{**" and followed by a "**}**".

NOTE – "**{**" and "**}**" are the names of lexical items containing the single characters **{** and **}** (see 12.37).

5.5.2 In this example, "IdentifierList" would be defined by a further production, either before or after the production defining "ExampleProduction".

5.6 Layout

Each production used in this Recommendation | International Standard is preceded and followed by an empty line. Empty lines do not appear within productions. The production may be on a single line, or may be spread over several lines. Layout is not significant.

© ISO/IEC 2021 – All rights reserved
Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

5.7 Recursion

The productions in this Recommendation | International Standard are frequently recursive. In this case the productions are to be continuously reapplied until no new sequences are generated.

NOTE – In many cases, such reapplication results in an infinite set of permitted sequences of lexical items. Some or all of the sequences in the set may themselves contain an unbounded number of lexical items. This is not an error.

5.8 References to permitted sequences of lexical items

This Recommendation | International Standard references a permitted sequence of lexical items (part of the ASN.1 notation) by referencing the name that appears before the "::=" in a production; the name is surrounded by the QUOTATION MARK (34) character (") to distinguish it from natural language text, unless it appears as part of a production.

5.9 References to a lexical item

This Recommendation | International Standard references a lexical item by using the name of the lexical item; when the name appears in natural language text, and could be confused with such text, then it is surrounded by the QUOTATION MARK (34) character (").

5.10 Short-hand notations

In order to make productions more concise and more readable, the following short-hand notations are used in the definition of permitted sequences of lexical items in this Recommendation | International Standard and also in Rec. ITU-T X.681 | ISO/IEC 8824-2, Rec. ITU-T X.682 | ISO/IEC 8824-3 and Rec. ITU-T X.683 | ISO/IEC 8824-4:

a) An asterisk (*) following two names, "A" and "B", denotes the "empty" lexical item (see 12.7), or one of the permitted sequences of lexical items associated with "A", or an alternating series of one of the sequences of lexical items associated with "A" and one of the sequences of lexical items associated with "B", both starting and finishing with one associated with "A". Thus:

C ::= A B *

is equivalent to:

C ::= D | empty
D ::= A | A B D

"D" being an auxiliary name not appearing elsewhere in the productions.

EXAMPLE – "C ::= A B *" is the shorthand notation for the following alternatives of C:

empty
A
A B A
A B A B A
A B A B A B A
...

b) A plus sign (+) is similar to the asterisk in a), except that the "empty" lexical item is excluded. Thus:

E ::= A B +

is equivalent to:

E ::= A | A B E

EXAMPLE – "E ::= A B +" is the shorthand notation for the following alternatives of E:

A
A B A
A B A B A
A B A B A B A
...

c) A question mark (?) following a name denotes either the "empty" lexical item (see 12.7) or a permitted sequence of lexical items associated with "A". Thus:

F ::= A ?

© ISO/IEC 2021 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

is equivalent to:

F ::= empty | A

NOTE – These short-hand notations take precedence over the juxtaposition of lexical items in production sequences (see 5.2.2).

5.11 Value references and the typing of values

5.11.1 The ASN.1 value assignment notation enables a name to be given to a value of a specified type. This name can be used wherever a reference to that value is needed. Annex C describes and specifies the value mapping mechanism that allows a value reference name for a value of one type to identify a value of a second (similar) type. Thus, a reference to the first value can be used wherever a reference to a value in the second type is required.

5.11.2 In the body of the ASN.1 standards normal English text is used to specify legality (or otherwise) of constructs where more than one type is involved. These legality specifications generally require that two or more types be "compatible". For example, the type used in defining a value reference is required to be "compatible with" the governing type when the value reference is used. The normative Annex C uses the value mapping concept to give a precise statement about whether any given ASN.1 construct is legal or not.

6 The ASN.1 model of type extension

When decoding an extensible type, a decoder may detect:

a) the absence of expected extension additions in a sequence or set type; or

b) the presence of arbitrary unexpected extension additions above those defined (if any) in a sequence or set type, or of an unknown alternative in a choice type, or an unknown enumeration in an enumerated type, or of an unexpected length or value of a type whose constraint is extensible.

In formal terms, an abstract syntax defined by the extensible type **X** contains not only the values of type **X**, but also the values of all types that are extension-related to **X**. Thus, the decoding process never signals an error when either of the above situations (a or b) is detected. The action that is taken in each situation is determined by the ASN.1 specifier.

NOTE – Frequently the action will be to ignore the presence of unexpected additional extensions, and to use a default value or a "missing" indicator for expected extension additions that are absent.

Unexpected extension additions detected by a decoder in an extensible type can later be included in a subsequent encoding of that type (for transmission back to the sender, or to some third party), provided that the same transfer syntax is used on the subsequent transmission.

7 Extensibility requirements on encoding rules

NOTE – These requirements apply to standardized encoding rules. They do not apply to encoding rules defined using ECN (see Rec. ITU-T X.692 | ISO/IEC 8825-3).

7.1 All ASN.1 encoding rules shall allow the encoding of values of an extensible type **X** in such a way that they can be decoded using an extensible type **Y** that is extension-related to **X**. Further, the encoding rules shall allow the values that were decoded using **Y** to be re-encoded (using **Y**) and decoded using a third extensible type **Z** that is extension related to **Y** (and hence **X** also).

NOTE – Types **X**, **Y** and **Z** may appear in any order in the extension series.

If a value of an extensible type **X** is encoded and then relayed (directly or through a relaying application using extension-related type **Z**) to another application that decodes the value using extensible type **Y** that is extension-related to **X**, then the decoder using type **Y** obtains an abstract value composed of:

a) an abstract value of the extension root type;

b) an abstract value of each extension addition that is present in both **X** and **Y**;

c) delimited encoding for each extension addition (if any) that is in **X** but not in **Y**.

The encodings in c) shall be capable of being included in a later encoding of a value of **Y**, if so required by the application. That encoding shall be a valid encoding of a value of **X**.

Tutorial example: If system A is using an extensible root type (type **X**) that is a sequence type or a set type with an extension addition of an optional integer type, while system B is using an extension-related type (type **Y**) that has two extension additions where each is an optional integer type, then transmission by B of a value of **Y** which omits the integer value of the first extension addition and includes the second must not be confused by A with the presence of the first (only) extension addition of **X** that it knows about. Moreover, A must be able to re-encode the value of **X** with a

© ISO/IEC 2021 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

value present for the first integer type, followed by the second integer value received from B, if so required by the application protocol.

7.2 All ASN.1 encoding rules shall specify the encoding and decoding of the value of an enumerated type and a choice type in such a way that if a transmitted value is in the set of extension additions held in common by the encoder and the decoder, then it is successfully decoded; otherwise, it shall be possible for the decoder to delimit the encoding of it and to identify it as a value of an (unknown) extension addition.

7.3 All ASN.1 encoding rules shall specify the encoding and decoding of types with extensible constraints in such a way that if a transmitted value is in the set of extension additions held in common by the encoder and the decoder, then it is successfully decoded, otherwise it shall be possible for the decoder to delimit the encoding of and to identify it as a value of an (unknown) extension addition.

In all cases, the presence of extension additions shall not affect the ability to recognize later material when a type with an extension marker is nested inside some other type.

NOTE 1 – All variants of the Basic Encoding Rules of ASN.1 and the Packed Encoding Rules of ASN.1 satisfy all these requirements. Encoding rules defined using ECN do not necessarily satisfy all these requirements, but may do so.

NOTE 2 – PER and BER do not identify the version number in the encoding of an extension addition. Encodings specified using ECN may or may not provide such identification.

8 Tags

8.1 A tag is specified (either within the text of this Recommendation | International Standard or by using a type prefix) by giving a class and a number within the class. The class is one of:

- universal;
- application;
- private;
- context-specific.

8.2 The number is a non-negative integer, specified in decimal notation.

8.3 Restrictions on tags assigned by the user of ASN.1 are specified in 31.2.

NOTE – Subclause 31.2 includes the restriction that users of this notation are not allowed to explicitly specify universal class tags in their ASN.1 specifications. There is no formal difference between use of tags from the other three classes. Where application class tags are employed, a private or context-specific class tag could generally be applied instead, as a matter of user choice and style. The presence of the three classes is largely for historical reasons, but guidance is given in G.2.12 on the way in which the classes are usually employed.

8.4 Table 1 summarizes the assignment of tags in the universal class which are specified in this Recommendation | International Standard.

8.5 Some encoding rules require a canonical order for tags. To provide uniformity, a canonical order for tags is defined in 8.6.

8.6 The canonical order for tags is based on the outermost tag of each type and is defined as follows:

a) those elements or alternatives with universal class tags shall appear first, followed by those with application class tags, followed by those with context-specific tags, followed by those with private class tags;

b) within each class of tags, the elements or alternatives shall appear in ascending order of their tag numbers.

© ISO/IEC 2021 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

Table 1 – Universal class tag assignments

UNIVERSAL 0	Reserved for use by the encoding rules
UNIVERSAL 1	Boolean type
UNIVERSAL 2	Integer type
UNIVERSAL 3	Bitstring type
UNIVERSAL 4	Octetstring type
UNIVERSAL 5	Null type
UNIVERSAL 6	Object identifier type
UNIVERSAL 7	Object descriptor type
UNIVERSAL 8	External type and Instance-of type
UNIVERSAL 9	Real type
UNIVERSAL 10	Enumerated type
UNIVERSAL 11	Embedded-pdv type
UNIVERSAL 12	UTF8String type
UNIVERSAL 13	Relative object identifier type
UNIVERSAL 14	The time type
UNIVERSAL 15	Reserved for future editions of this Recommendation | International Standard
UNIVERSAL 16	Sequence and Sequence-of types
UNIVERSAL 17	Set and Set-of types
UNIVERSAL 18-22, 25-30	Character string types
UNIVERSAL 23-24	**`UTCTime`** and **`GeneralizedTime`**
UNIVERSAL 31-34	**`DATE`**, **`TIME-OF-DAY`**, **`DATE-TIME`** and **`DURATION`** respectively
UNIVERSAL 35	OID internationalized resource identifier type
UNIVERSAL 36	Relative OID internationalized resource identifier type
UNIVERSAL 37-...	Reserved for addenda to this Recommendation | International Standard

9 Encoding instructions

9.1 An encoding instruction is assigned to a type using either a type prefix (see 31.3) or an encoding control section (see clause 54).

9.2 A type prefix may contain an encoding reference. If it does not, the encoding reference is determined by the default encoding reference for the module (see 13.5).

9.3 An encoding control section always contains an encoding reference. There may be multiple encoding control sections, but each encoding control section shall have a distinct encoding reference.

9.4 An encoding instruction consists of a sequence of lexical items specified in the Recommendation | International Standard determined by the encoding reference (see Annex E).

9.5 Multiple encoding instructions with the same or with different encoding references may be assigned to a type (using either or both of type prefixes and an encoding control section). Encoding instructions assigned with a given encoding reference are independent from those assigned with a different encoding reference, and from any use of a type prefix to perform tagging.

9.6 The effect of assigning several encoding instructions with the same encoding reference (using either or both of type prefixes and an encoding control section) is specified in the Recommendation | International Standard determined by the encoding reference (see Annex E), and is not specified in this Recommendation | International Standard.

9.7 If an encoding instruction is assigned to the "Type" in a "TypeAssignment", it becomes associated with the type, and is applied wherever the "typereference" of the "TypeAssignment" is used. This includes use in other modules through the export and import statements.

© ISO/IEC 2021 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

10 Use of the ASN.1 notation

10.1 The ASN.1 notation for a type definition shall be "Type" (see 17.1).

10.2 The ASN.1 notation for a value of a type shall be "Value" (see 17.7).

NOTE – It is not in general possible to interpret the value notation without knowledge of the type.

10.3 The ASN.1 notation for assigning a type to a type reference name shall be either "TypeAssignment" (see 16.1), "ValueSetTypeAssignment" (see 16.6), "ParameterizedTypeAssignment" (see Rec. ITU-T X.683 | ISO/IEC 8824-4, 8.2), or "ParameterizedValueSetTypeAssignment" (see Rec. ITU-T X.683 | ISO/IEC 8824-4, 8.2).

10.4 The ASN.1 notation for assigning a value to a value reference name shall be either "ValueAssignment" (see 16.2) or "ParameterizedValueAssignment" (see Rec. ITU-T X.683 | ISO/IEC 8824-4, 8.2).

10.5 The production alternatives of the notation "Assignment" shall only be used within the notation "ModuleDefinition" (except as specified in NOTE 2 of 13.1).

11 The ASN.1 character set

11.1 A lexical item shall consist of a sequence of the characters listed in Table 2 except as specified in 11.2, 11.3 and 11.4. In Table 2, characters are identified by the names they are given in ISO/IEC 10646.

Table 2 – ASN.1 characters

A to Z	(LATIN CAPITAL LETTER A to LATIN CAPITAL LETTER Z)
a to z	(LATIN SMALL LETTER A to LATIN SMALL LETTER Z)
0 to 9	(DIGIT ZERO to DIGIT 9)
!	(EXCLAMATION MARK)
"	(QUOTATION MARK)
&	(AMPERSAND)
'	(APOSTROPHE)
(	(LEFT PARENTHESIS)
)	(RIGHT PARENTHESIS)
*	(ASTERISK)
,	(COMMA)
-	(HYPHEN-MINUS)
.	(FULL STOP)
/	(SOLIDUS)
:	(COLON)
;	(SEMICOLON)
<	(LESS-THAN SIGN)
=	(EQUALS SIGN)
>	(GREATER-THAN SIGN)
@	(COMMERCIAL AT)
[	(LEFT SQUARE BRACKET)
]	(RIGHT SQUARE BRACKET)
^	(CIRCUMFLEX ACCENT)
_	(LOW LINE)
{	(LEFT CURLY BRACKET)
|	(VERTICAL LINE)
}	(RIGHT CURLY BRACKET)
–	(NON-BREAKING HYPHEN)

NOTE – Where equivalent derivative standards are developed by national standards bodies, additional characters may appear in the following lexical items:

- typereference (see 12.2);
- identifier (see 12.3);
- valuereference (see 12.4);
- modulereference (see 12.5).

© ISO/IEC 2021 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

When additional characters are introduced to accommodate a language in which the distinction between upper-case and lower-case letters is without meaning, the syntactic distinction achieved by dictating the case of the first character of certain of the above lexical items has to be achieved in some other way. This is to allow valid ASN.1 specifications to be written in various languages.

11.2 Where the notation is used to specify the value of a character string type, all characters for the defined character set can appear in the ASN.1 notation, surrounded by the QUOTATION MARK (34) characters (") (see 12.14).

11.3 Additional (arbitrary) graphic symbols may appear in the "comment" lexical item (see 12.6).

11.4 Where the notation is used to specify the value of a Unicode label, all characters allowed in a Unicode label can appear in ASN.1 notation.

11.5 There shall be no significance placed on the typographical style, size, colour, intensity, or other display characteristics.

11.6 The upper-case and lower-case letters shall be regarded as distinct.

11.7 ASN.1 definitions can also contain white-space characters (see 12.1.6) between lexical items.

11.8 The NON-BREAKING HYPHEN and the HYPHEN-MINUS should be treated as identical in all names (including reserved words).

NOTE – A name such as My-Type is the same name whether it contains a HYPHEN-MINUS or a NON-BREAKING HYPHEN.

12 ASN.1 lexical items

12.1 General rules

12.1.1 The following subclauses specify the characters in lexical items. In each case the name of the lexical item is given, together with the definition of the character sequences which form the lexical item.

12.1.2 The lexical items specified in the subclauses of this clause 12 (except multiple-line "comment", "bstring", "xmlbstring", "hstring", "xmlhstring", "cstring", "xmlcstring" and "simplestring") shall not contain white-space (see 12.6, 12.10, 12.11, 12.12, 12.13, 12.14, 12.15 and 12.16). A single-line "comment" shall not contain "new-line" characters. The "non-integerUnicodeLabel" lexical item may include the NON-BREAKING-SPACE character.

12.1.3 The length of a line is not restricted.

12.1.4 Lexical items may be separated by one or more occurrences of white-space (see 12.1.6) or comments (see 12.6) except when the non-spacing indicator "**&**" (see 5.4) is used. Within an "XMLTypedValue" production (see 16.2), white-space may appear between lexical items, but the "comment" lexical item shall not be present.

NOTE – This is to avoid ambiguity resulting from the presence of adjacent hyphens or asterisk and solidus within an "xmlcstring" lexical item. Such characters never indicate the start of a "comment" lexical item when they appear within an "XMLTypedValue" production.

12.1.5 A lexical item shall be separated from a following lexical item by one or more instances of white-space or comment if the initial character (or characters) of the following lexical item is a permitted character (or characters) for inclusion at the end of the characters in the earlier lexical item.

12.1.6 This Recommendation | International Standard uses the terms "newline", and "white-space". In representing white-space and newline (end of line) in machine-readable specifications, any one or more of the following characters may be used in any combination (for each character, the character name and character code specified in The Unicode Standard are given):

For white-space:

HORIZONTAL TABULATION (9)

LINE FEED (10)

VERTICAL TABULATION (11)

FORM FEED (12)

CARRIAGE RETURN (13)

SPACE (32)

NO-BREAK SPACE ({0,0,0,160})

© ISO/IEC 2021 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

For newline:

LINE FEED (10)

VERTICAL TABULATION (11)

FORM FEED (12)

CARRIAGE RETURN (13)

NOTE – Any character or character sequence that is a valid newline is also a valid white-space.

12.2 Type references

Name of lexical item – typereference

12.2.1 A "typereference" shall consist of an arbitrary number (one or more) of letters, digits, and hyphens. The initial character shall be an upper-case letter. A hyphen shall not be the last character. A hyphen shall not be immediately followed by another hyphen.

NOTE – The rules concerning hyphen are designed to avoid ambiguity with (possibly following) comment.

12.2.2 A "typereference" shall not be one of the reserved character sequences listed in 12.38.

12.3 Identifiers

Name of lexical item – identifier

An "identifier" shall consist of an arbitrary number (one or more) of letters, digits, and hyphens. The initial character shall be a lower-case letter. A hyphen shall not be the last character. A hyphen shall not be immediately followed by another hyphen.

NOTE – The rules concerning hyphen are designed to avoid ambiguity with (possibly following) comment.

12.4 Value references

Name of lexical item – valuereference

A "valuereference" shall consist of the sequence of characters specified for an "identifier" in 12.3. In analyzing an instance of use of this notation, a "valuereference" is distinguished from an "identifier" by the context in which it appears.

12.5 Module references

Name of lexical item – modulereference

A "modulereference" shall consist of the sequence of characters specified for a "typereference" in 12.2. In analyzing an instance of use of this notation, a "modulereference" is distinguished from a "typereference" by the context in which it appears.

12.6 Comments

Name of lexical item – comment

12.6.1 A "comment" is not referenced in the definition of the ASN.1 notation. It may, however, appear at any time between other lexical items, and has no syntactic significance.

NOTE – Nonetheless, in the context of a Recommendation | International Standard that uses ASN.1, an ASN.1 comment may contain normative text related to the application semantics, or constraints on the syntax.

12.6.2 The lexical item "comment" can have two forms:

a) One-line comments which begin with "`--`" as defined in 12.6.3;

b) Multiple-line comments which begin with "`/*`" as defined in 12.6.4.

12.6.3 Whenever a "comment" begins with a pair of adjacent hyphens, it shall end with the next pair of adjacent hyphens or at the end of the line, whichever occurs first. A comment shall not contain a pair of adjacent hyphens other than the pair which starts it and the pair, if any, which ends it. If a comment beginning with "`--`" includes the adjacent characters "`/*`" or "`*/`", these have no special meaning and are considered part of the comment. The comment may include graphic symbols which are not in the character set specified in 11.1 (see 11.3).

© ISO/IEC 2021 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

12.6.4 Whenever a "comment" begins with "`/*`", it shall end with a corresponding "`*/`", whether this "`*/`" is on the same line or not. If another "`/*`" is found before a "`*/`", then the comment terminates when a matching "`*/`" has been found for each "`/*`". If a comment beginning with "`/*`" includes two adjacent hyphens "`--`", these hyphens have no special meaning and are considered part of the comment. The comment may include graphic symbols which are not in the character set specified in 11.1 (see 11.3).

NOTE – This allows the user to comment parts of an ASN.1 module that already contain comments (whether they begin with "`--`" or "`/*`") by simply inserting "`/*`" at the beginning of the part to be commented and "`*/`" at its end, provided there are no character string values within the part to be commented out that contain "`/*`" or "`*/`".

12.7 Empty lexical item

Name of lexical item – empty

The "empty" item contains no characters. It is used in the notation of clause 5 when alternative sets of production sequences are specified, to indicate that absence of all alternatives is possible.

12.8 Numbers

Name of lexical item – number

A "number" shall consist of one or more digits. The first digit shall not be zero unless the "number" is a single digit.

NOTE – The "number" lexical item is always mapped to an integer value by interpreting it as decimal notation.

12.9 Real numbers

Name of lexical item – realnumber

A "realnumber" shall consist of an integer part that is a series of one or more digits, and optionally a decimal point (.). The decimal point can optionally be followed by a fractional part which is one or more digits. The integer part, decimal point or fractional part (whichever is last present) can optionally be followed by an e or E and an optionally-signed exponent which is one or more digits. The leading digit of the "realnumber" shall not be zero unless it is either the only digit or is immediately followed by a decimal point followed by a fractional part of which at least one digit is not zero.

A "number" is also a valid instance of "realnumber". In analyzing an instance of use of this notation, a "realnumber" is distinguished from a "number" by the context in which it appears.

12.10 Binary strings

Name of lexical item – bstring

A "bstring" shall consist of an arbitrary number (possibly zero) of the characters:

`0 1`

possibly intermixed with white-space, preceded by an APOSTROPHE (39) character (`'`) and followed by the pair of characters:

`'B`

EXAMPLE – `'01101100'B`

Occurrences of white-space within a binary string lexical item have no significance.

12.11 XML binary string item

Name of item – xmlbstring

An "xmlbstring" shall consist of an arbitrary number (possibly zero) of zeros, ones or white-space. Any white-space characters that appear within a binary string item have no significance.

EXAMPLE – 01101100

This sequence of characters is also a valid instance of "xmlhstring" and "xmlcstring". In analyzing an instance of use of this notation, an "xmlbstring" is distinguished from an "xmlhstring" or "xmlcstring" by the context in which it appears.

12.12 Hexadecimal strings

Name of lexical item – hstring

© ISO/IEC 2021 – All rights reserved
Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

12.12.1 An "hstring" shall consist of an arbitrary number (possibly zero) of the characters:

A B C D E F 0 1 2 3 4 5 6 7 8 9

possibly intermixed with white-space, preceded by an APOSTROPHE (39) character (**'**) and followed by the pair of characters:

'H

EXAMPLE – **'AB0196'H**

Occurrences of white-space within a hexadecimal string lexical item have no significance.

12.12.2 Each character is used to denote the value of a semi-octet using a hexadecimal representation.

12.13 XML hexadecimal string item

Name of item – xmlhstring

12.13.1 An "xmlhstring" shall consist of an arbitrary number (possibly zero) of the characters:

0 1 2 3 4 5 6 7 8 9 A B C D E F a b c d e f

or white-space. Any white-space characters that appear within a hexadecimal string item have no significance.

EXAMPLE – Ab0196

12.13.2 Each character is used to denote the value of a semi-octet using a hexadecimal representation.

12.13.3 All instances of "xmlhstring" are also valid instances of "xmlcstring", and some instances are also valid instances of "xmlbstring". In analyzing an instance of use of this notation, an "xmlhstring" is distinguished from an "xmlbstring" or "xmlcstring" by the context in which it appears.

12.14 Character strings

Name of lexical item – cstring

12.14.1 A "cstring" shall consist of an arbitrary number (possibly zero) of graphic symbols and spacing characters from the character set referenced by the character string type, preceded and followed by a QUOTATION MARK (34) character ("). If the character set includes a QUOTATION MARK (34) character, this character (if present in the character string being represented by the "cstring") shall be represented in the "cstring" by a pair of QUOTATION MARK (34) characters on the same line with no intervening spacing character. The "cstring" may span more than one line of text, in which case the character string being represented shall not include spacing characters in the position prior to or following the end of line in the "cstring". Any spacing characters that appear immediately prior to or following the end of line in the "cstring" have no significance.

NOTE 1 – The "cstring" can only be used to unambiguously represent (on a printed page) character strings for which every character in the string being represented has either been assigned a graphic symbol, or is a spacing character. Where a character string containing control characters needs to be denoted in a printed representation, alternative ASN.1 syntax is available (see clause 39).

NOTE 2 – The character string represented by a "cstring" consists of the characters associated with the graphic symbols and spacing characters. Spacing characters immediately preceding or following any end of line in the "cstring" are not part of the character string being represented (they are ignored). Where spacing characters are included in the "cstring", or where the graphic symbols in the character repertoire are not unambiguous in a printed representation, the character string denoted by "cstring" may be ambiguous in that printed representation.

EXAMPLE 1 – "屎 屍 市 弑"

EXAMPLE 2 – The "cstring":

"ABCDE FGH

IJK""XYZ"

can be used to represent a character string value of type **IA5String**. The value represented consists of the characters:

ABCDE FGHIJK"XYZ

where the precise number of spaces intended between **E** and **F** can be ambiguous in a printed representation if a proportional spacing font (such as is used above) is used in the printed specification, or if the character repertoire contains multiple spacing characters of different widths.

© ISO/IEC 2021 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

12.14.2 When a character is a combining character (see Annex H) it shall be denoted in a printed representation of the "cstring" as an individual character. It shall not be overprinted with the characters with which it combines. (This ensures that the order of combining characters in the string value is unambiguously defined in the printed version.)

EXAMPLE – Lower-case "**e**" and the accent combining character are two characters in ISO/IEC 10646, and thus a corresponding "cstring" should be printed as two characters and not as the single character **é**.

12.15 XML character string item

Name of item – xmlcstring

12.15.1 An "xmlcstring" shall consist of an arbitrary number (possibly zero) of the following ISO/IEC 10646 characters:

a) HORIZONTAL TABULATION (9);

b) LINE FEED (10);

c) CARRIAGE RETURN (13);

d) any character whose ISO/IEC 10646 character code is in the range 32 (20 hex) to 55295 (D7FF hex), inclusive;

e) any character whose ISO/IEC 10646 character code is in the range 57344 (E000 hex) to 65533 (FFFD hex), inclusive;

f) any character whose ISO/IEC 10646 character code is in the range 65536 (10000 hex) to 1114111 (10FFFF hex), inclusive.

NOTE – Additional restrictions are imposed by the requirement that the "xmlcstring", in an instance of use, shall contain only characters permitted by the governing character string type.

12.15.2 The characters "&" (AMPERSAND), "<" (LESS-THAN SIGN) or ">" (GREATER-THAN SIGN) shall appear only as part of one of the character sequences specified in 12.15.4 or 12.15.5.

12.15.3 An "xmlcstring" is used to represent the value of a restricted character string (see 41.9), and can be used to represent all combinations of ISO/IEC 10646 characters, either directly, or by using the escape sequences specified below.

NOTE 1 – An "xmlcstring" cannot be used to represent characters that are not present in ISO/IEC 10646, such as some of the control characters which can appear in **`GeneralString`**, nor can it represent characters which might be defined with ISO/IEC 10646 character codes above 10FFFF hex.

NOTE 2 – The characters LINE FEED (10) and CARRIAGE RETURN (13) and the pair CARRIAGE RETURN + LINE FEED are not distinguished when processed by conforming XML processors.

12.15.4 If the characters "&" (AMPERSAND), "<" (LESS-THAN SIGN) or ">" (GREATER-THAN SIGN) are present in an abstract character string value being represented by "xmlcstring" (see 41.9), they shall be represented in the "xmlcstring" by either

a) the escape sequences specified in 12.15.8; or

b) the escape sequences "&", "<" or ">" respectively. These escape sequences shall not contain white-space (see 12.1.6).

12.15.5 If a character with an ISO/IEC 10646 character code in column 1 of Table 3 is present in the abstract character string value being represented by the "xmlcstring" (see 41.9), it shall be represented by the character sequence in column 2 of Table 3. These character sequences shall not contain white-space (see 12.1.6).

NOTE – This does not include characters with decimal character codes 9, 10, and 13, and all the letters in these character sequences are lower-case.

© ISO/IEC 2021 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

Table 3 – Escape sequences for control characters in an "xmlcstring"

ISO/IEC 10646 character code	"xmlcstring" representation	ISO/IEC 10646 character code	"xmlcstring" representation
0 (0 hex)	<nul/>	17 (11 hex)	<dc1/>
1 (1 hex)	<soh/>	18 (12 hex)	<dc2/>
2 (2 hex)	<stx/>	19 (13 hex)	<dc3/>
3 (3 hex)	<etx/>	20 (14 hex)	<dc4/>
4 (4 hex)	<eot/>	21 (15 hex)	<nak/>
5 (5 hex)	<enq/>	22 (16 hex)	<syn/>
6 (6 hex)	<ack/>	23 (17 hex)	<etb/>
7 (7 hex)	<bel/>	24 (18 hex)	<can/>
8 (8 hex)	<bs/>	25 (19 hex)	
11 (B hex)	<vt/>	26 (1A hex)	<sub/>
12 (C hex)	<ff/>	27 (1B hex)	<esc/>
14 (E hex)	<so/>	28 (1C hex)	<is4/>
15 (F hex)	<si/>	29 (1D hex)	<is3/>
16 (10 hex)	<dle/>	30 (1E hex)	<is2/>
		31 (1F hex)	<is1/>

12.15.6 When "xmlcstring" is used within an "XMLTypedValue" (see 16.2) forming part of an XER encoding (see Rec. ITU-T X.693 | ISO/IEC 8825-4), it may contain adjacent HYPHEN-MINUS (45) characters. When used within an instance of XML value notation in an ASN.1 module, it shall not contain two adjacent HYPHEN-MINUS characters. If this character sequence is present in an abstract character string value being represented by the "xmlcstring" in an ASN.1 module, then at least one of the adjacent HYPHEN-MINUS characters shall be represented by the escape sequences specified in 12.15.8.

12.15.7 When "xmlcstring" is used within an "XMLTypedValue" forming part of an XER encoding (see Rec. ITU-T X.693 | ISO/IEC 8825-4), it may contain adjacent ASTERISK (42) and SOLIDUS (47) characters in any order. When used within an instance of XML value notation in an ASN.1 module, it shall not contain adjacent ASTERISK and SOLIDUS characters (in any order). If this character sequence is present in an abstract character string value being represented by the "xmlcstring", then at least one of the adjacent ASTERISK and SOLIDUS characters shall be represented by the escape sequences specified in 12.15.8.

12.15.8 Any character that can appear directly in an "xmlcstring" can also be represented in the "xmlcstring" by an escape sequence of the form "&#n;" (where n is the ISO/IEC 10646 character code in decimal notation) or of the form "&#xn;" (where n is the ISO/IEC 10646 character code in hexadecimal notation). These escape sequences shall not contain white-space (see 12.1.6).

NOTE 1 – Leading zeros are permitted in the decimal and hexadecimal values of "n" and both lower-case and upper-case letters "A"-"F" can be used in the hexadecimal value.

NOTE 2 – If the escape sequences "&#n" and "&#xn" are used for ISO/IEC 10646 characters which are not in the Basic Multilingual Plane (BMP), the value of "n" will be greater than 65535 (FFFF hex).

EXAMPLE – The "xmlcstring":

ABCDé FGHîJK&XYZ

can be used to represent a character string value of type UTF8String. The value represented consists of the characters:

ABCDé FGHîJK&XYZ

where the precise space characters between é and F can be ambiguous in print media if a proportional spacing font (such as above) is used in the specification.

© ISO/IEC 2021 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

12.16 The simple character string lexical item

Name of item – simplestring

A "simplestring" shall consist of one or more ISO/IEC 10646 characters whose character code is in the range 32 to 126, preceded and followed by a QUOTATION MARK (34) character ("). It shall not contain a QUOTATION MARK (34) character ("). The "simplestring" may span more than one line of text, in which case any characters representing end-of-line shall be treated as spacing characters. In analyzing an instance of use of this notation, a "simplestring" is distinguished from a "cstring" by the context in which it appears.

NOTE – The "simplestring" lexical item is only used in the subtype notation of the time type.

12.17 Time value character strings

Name of item – tstring

A "tstring" shall consist of one or more of the characters:

```
0 1 2 3 4 5 6 7 8 9 + - : . , / C D H M R P S T W Y Z
```

preceded and followed by a QUOTATION MARK (34) character ("").

NOTE – The "tstring" lexical item is only used in the value notation for the time type.

12.18 XML time value character string item

Name of item – xmltstring

An "xmltstring" shall consist of one or more of the characters:

```
0 1 2 3 4 5 6 7 8 9 + - : . , / C D H M R P S T W Y Z
```

NOTE – The "xmltstring" lexical item is only used in the XML value notation of the time type.

12.19 The property and setting names lexical item

Name of item – psname

A "psname" shall consist of an arbitrary number (one or more) of letters, digits and hyphens. The initial character shall be an upper-case letter. A hyphen shall not be the last character. A hyphen shall not be immediately followed by another hyphen.

NOTE – The "psname" lexical item is only used in the contents of the "simplestring" used in the subtype notation for the time type.

12.20 Assignment lexical item

Name of lexical item – "`::=`"

This lexical item shall consist of the sequence of characters:

```
::=
```

NOTE – This sequence does not contain white-space (see 12.1.2).

12.21 Range separator

Name of lexical item – "`..`"

This lexical item shall consist of the sequence of characters:

```
..
```

NOTE – This sequence does not contain white-space (see 12.1.2).

12.22 Ellipsis

Name of lexical item – "`...`"

This lexical item shall consist of the sequence of characters:

```
...
```

NOTE – This sequence does not contain white-space (see 12.1.2).

© ISO/IEC 2021 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

12.23 Left version brackets

Name of lexical item – "`[[`"

This lexical item shall consist of the sequence of characters:

`[[`

NOTE – This sequence does not contain white-space (see 12.1.2).

12.24 Right version brackets

Name of lexical item – "`]]`"

This lexical item shall consist of the sequence of characters:

`]]`

NOTE – This sequence does not contain white-space (see 12.1.2).

12.25 Encoding references

Name of item – encodingreference

An "encodingreference" shall consist of a sequence of characters as specified for a "typereference" in 12.2, except that no lower-case letters shall be included.

NOTE – Currently defined encoding references are listed in Annex E with the Recommendation | International Standard that specifies the syntax and semantics of the corresponding encoding instructions. The "encodingreference" shall consist only of the sequences listed in Annex E in this or in future versions of this Recommendation | International Standard.

12.26 Integer-valued Unicode labels

Name of lexical item – integerUnicodeLabel

This lexical item shall consist of an arbitrarily long sequence of ISO/IEC 10646 characters in the range 0 (DIGIT ZERO) to 9 (DIGIT NINE) that identify an arc of the International Object Identifier tree. It shall not commence with a 0 (DIGIT ZERO) character unless it has only a single character and the primary integer value of the associated arc of the International Object Identifier tree is zero.

12.27 Non-integer Unicode labels

Name of lexical item – non-integerUnicodeLabel

This lexical item shall consist of an arbitrarily long sequence of ISO/IEC 10646 characters that satisfies the constraints specified in Rec. ITU-T X.660 | ISO 9834-1, 7.5 and identifies an arc of the International Object Identifier tree. For lexical parsing purposes, it shall not consist only of characters that would enable it to be identified as an "integerUnicodeLabel".

12.28 XML end tag start item

Name of item – "</"

This item shall consist of the sequence of characters:

</

NOTE – This sequence does not contain any white-space characters (see 12.1.2).

12.29 XML single tag end item

Name of item – "/>"

This item shall consist of the sequence of characters:

/>

NOTE – This sequence does not contain any white-space characters (see 12.1.2).

12.30 XML boolean true item

Name of item – "true"

© ISO/IEC 2021 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

12.30.1 This item shall consist of the sequence of characters:

true

12.30.2 In analyzing an instance of use of this notation, a "true" is distinguished from a "valuereference" or an "identifier" or an instance of XML boolean "extended-true" by the context in which it appears.

NOTE – This sequence does not contain any white-space characters (see 12.1.2).

12.31 XML boolean extended-true item

Name of item – extended-true

12.31.1 This item shall consist of either the sequence of characters:

true

or of the single character:

1 (DIGIT ONE)

12.31.2 In analyzing an instance of use of this notation, an "extended-true" is distinguished from a "valuereference" or an "identifier" or an instance of XML boolean "true" by the context in which it appears.

NOTE – This sequence does not contain any white-space characters (see 12.1.2).

12.32 XML boolean false item

Name of item – "false"

12.32.1 This item shall consist of the sequence of characters:

false

12.32.2 In analyzing an instance of use of this notation, a "false" is distinguished from a "valuereference" or an "identifier" or an instance of XML boolean "extended-false" by the context in which it appears.

NOTE – This sequence does not contain any white-space characters (see 12.1.2).

12.33 XML boolean extended-false item

Name of item – extended-false

12.33.1 This item shall consist of either the sequence of characters:

false

or of the single character:

0 (DIGIT ZERO)

12.33.2 In analyzing an instance of use of this notation, a "false" is distinguished from a "valuereference" or an "identifier" or an instance of XML boolean "false" by the context in which it appears.

NOTE – This sequence does not contain any white-space characters (see 12.1.2).

12.34 XML real not-a-number item

Name of item – "NaN"

12.34.1 This item shall consist of the sequence of characters:

NaN

12.34.2 In analyzing an instance of use of this notation, a "NaN" is distinguished from any other lexical item commencing with an upper-case letter by the context in which it appears.

NOTE – This sequence does not contain any white-space characters (see 12.1.2).

12.35 XML real infinity item

Name of item – "INF"

12.35.1 This item shall consist of the sequence of characters:

INF

© ISO/IEC 2021 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

12.35.2 In analyzing an instance of use of this notation, an "INF" is distinguished from any other lexical item commencing with an upper-case letter by the context in which it appears.

NOTE – This sequence does not contain any white-space characters (see 12.1.2).

12.36 XML tag names for ASN.1 types

Name of item – xmlasn1typename

12.36.1 This Recommendation | International Standard uses the item "xmlasn1typename" when ASN.1 built-in types are to be used as XML tag names.

12.36.2 Table 4 lists the character sequences that are to form the "xmlasn1typename" for each of the ASN.1 built-in types listed in 17.2. The ASN.1 built-in type is identified in column 1 of Table 4 by its production name. The character sequence which shall be used for "xmlasn1typename" is identified in column 2 of Table 4, with no white-space before or after these character sequences.

12.36.3 The "xmlasn1typename" for the "UsefulType"s (see 45.1) shall be the "typereference" used in their definition.

12.36.4 The character sequence in the "xmlasn1typename" item for the "ObjectClassFieldType" and for the "InstanceOfType" are specified in Rec. ITU-T X.681 | ISO/IEC 8824-2, 14.1 and Annex C.

12.36.5 If the ASN.1 built-in type is a "PrefixedType" then the type which determines the "xmlasn1typename" shall be "Type" in the "PrefixedType" (see 31.1.5). If this is itself a "PrefixedType", then this subclause 12.36.5 shall be recursively applied.

NOTE – The subclauses of 26.10 specify the "Type" to be used for a "SelectionType" and a "ConstrainedType".

Table 4 – Characters in xmlasn1typename

ASN.1 type production name	Characters in xmlasn1typename
BitStringType	BIT_STRING
BooleanType	BOOLEAN
ChoiceType	CHOICE
DateType	DATE
DateTimeType	DATE_TIME
DurationType	DURATION
EmbeddedPDVType	SEQUENCE
EnumeratedType	ENUMERATED
ExternalType	SEQUENCE
InstanceOfType	SEQUENCE
IntegerType	INTEGER
IRIType	OID_IRI
NullType	NULL
ObjectClassFieldType	*See Rec. ITU-T X.681 | ISO/IEC 8824-2, 14.10 and 14.11*
ObjectIdentifierType	OBJECT_IDENTIFIER
OctetStringType	OCTET_STRING
PrefixedType	*See 12.36.5*
RealType	REAL
RelativeIRIType	RELATIVE_OID_IRI
RelativeOIDType	RELATIVE_OID
RestrictedCharacterStringType	*The type name (e.g.* `IA5String`*)*
SequenceType	SEQUENCE
SequenceOfType	SEQUENCE_OF
SetType	SET
SetOfType	SET_OF
TimeType	TIME
TimeOfDayType	TIME_OF_DAY
UnrestrictedCharacterStringType	SEQUENCE

© ISO/IEC 2021 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

12.37 Single character lexical items

Names of lexical items –

"{"
"}"
"<"
">"
","
"."
"/"
"("
")"
"["
"]"
"-" (HYPHEN-MINUS)
":"
"="
"""" (QUOTATION MARK)
"'" (APOSTROPHE)
";"
"@"
"|"
"!"
"^"

A lexical item with any of the names listed above shall consist of the single character without the quotation marks.

12.38 Reserved words

Names of reserved words –

ABSENT	ENCODED	INTERSECTION	SEQUENCE
ABSTRACT-SYNTAX	ENCODING-CONTROL	ISO646String	SET
ALL	END	MAX	SETTINGS
APPLICATION	ENUMERATED	MIN	SIZE
AUTOMATIC	EXCEPT	MINUS-INFINITY	STRING
BEGIN	EXPLICIT	NOT-A-NUMBER	SYNTAX
BIT	EXPORTS	NULL	T61String
BMPString	EXTENSIBILITY	NumericString	TAGS
BOOLEAN	EXTERNAL	OBJECT	TeletexString
BY	FALSE	ObjectDescriptor	TIME
CHARACTER	FROM	OCTET	TIME-OF-DAY
CHOICE	GeneralizedTime	OF	TRUE
CLASS	GeneralString	OID-IRI	TYPE-IDENTIFIER
COMPONENT	GraphicString	OPTIONAL	UNION
COMPONENTS	IA5String	PATTERN	UNIQUE
CONSTRAINED	IDENTIFIER	PDV	UNIVERSAL
CONTAINING	IMPLICIT	PLUS-INFINITY	UniversalString
DATE	IMPLIED	PRESENT	UTCTime
DATE-TIME	IMPORTS	PrintableString	UTF8String
DEFAULT	INCLUDES	PRIVATE	VideotexString
DEFINITIONS	INSTANCE	REAL	VisibleString
DURATION	INSTRUCTIONS	RELATIVE-OID	WITH
EMBEDDED	INTEGER	RELATIVE-OID-IRI	

© ISO/IEC 2021 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

Lexical items with the above names shall consist of the sequence of characters in the name, and are reserved character sequences.

NOTE 1 – White-space does not occur in these sequences.

NOTE 2 – The keywords **CLASS**, **CONSTRAINED**, **CONTAINING**, **ENCODED**, **INSTANCE**, **SYNTAX** and **UNIQUE** are not used in this Recommendation | International Standard; they are used in Rec. ITU-T X.681 | ISO/IEC 8824-2, Rec. ITU-T X.682 | ISO/IEC 8824-3 and Rec. ITU-T X.683 | ISO/IEC 8824-4.

13 Module definition

13.1 A "ModuleDefinition" is specified by the following productions:

```
ModuleDefinition ::=
    ModuleIdentifier
    DEFINITIONS
    EncodingReferenceDefault
    TagDefault
    ExtensionDefault
    "::="
    BEGIN
    ModuleBody
    EncodingControlSections
    END

ModuleIdentifier ::=
    modulereference
    DefinitiveIdentification

DefinitiveIdentification ::=
    DefinitiveOID
|   DefinitiveOIDandIRI
|   empty

DefinitiveOID ::=
    "{" DefinitiveObjIdComponentList "}"

DefinitiveOIDandIRI ::=
        DefinitiveOID
    IRIValue

DefinitiveObjIdComponentList ::=
    DefinitiveObjIdComponent
|   DefinitiveObjIdComponent DefinitiveObjIdComponentList

DefinitiveObjIdComponent ::=
    NameForm
|   DefinitiveNumberForm
|   DefinitiveNameAndNumberForm

DefinitiveNumberForm    ::= number

DefinitiveNameAndNumberForm ::= identifier "(" DefinitiveNumberForm ")"

EncodingReferenceDefault ::=
        encodingreference INSTRUCTIONS
    |   empty

TagDefault ::=
    EXPLICIT TAGS
|   IMPLICIT TAGS
|   AUTOMATIC TAGS
|   empty

ExtensionDefault ::=
    EXTENSIBILITY IMPLIED
|   empty

ModuleBody ::=
    Exports Imports AssignmentList
```

© ISO/IEC 2021 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

```
|       empty

Exports ::=
        EXPORTS SymbolsExported ";"
|       EXPORTS ALL ";"
|       empty

SymbolsExported ::=
        SymbolList
|       empty

Imports ::=
        IMPORTS SymbolsImported ";"
|       empty

SymbolsImported ::=
        SymbolsFromModuleList
|       empty

SymbolsFromModuleList ::=
        SymbolsFromModule
|       SymbolsFromModuleList  SymbolsFromModule

SymbolsFromModule ::=
        SymbolList FROM GlobalModuleReference SelectionOption

SelectionOption ::=
        empty
|       WITH "SUCCESSORS"
|       WITH "DESCENDANTS"

GlobalModuleReference ::=
        modulereference AssignedIdentifier

AssignedIdentifier ::=
        ObjectIdentifierValue
|       DefinedValue
|       empty

SymbolList ::=
        Symbol
|       SymbolList "," Symbol

Symbol ::=
        Reference
|       ParameterizedReference

Reference ::=
        typereference
|       valuereference
|       objectclassreference
|       objectreference
|       objectsetreference

AssignmentList ::=
        Assignment
|       AssignmentList Assignment

Assignment ::=
        TypeAssignment
|       ValueAssignment
|       XMLValueAssignment
|       ValueSetTypeAssignment
|       ObjectClassAssignment
|       ObjectAssignment
|       ObjectSetAssignment
|       ParameterizedAssignment
```


© ISO/IEC 2021 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

NOTE 1 – The use of a "ParameterizedReference" in the "Exports" and "Imports" lists is specified in Rec. ITU-T X.683 | ISO/IEC 8824-4.

NOTE 2 – For examples (and for the definition in this Recommendation | International Standard of types with universal class tags), the "ModuleBody" can be used outside of a "ModuleDefinition".

NOTE 3 – "TypeAssignment", "ValueAssignment", "XMLValueAssignment" and "ValueSetTypeAssignment" productions are specified in clause 16.

NOTE 4 – The value of "TagDefault" for the module definition affects only those types defined explicitly in the module. It does not affect the interpretation of imported types.

NOTE 5 – The character semicolon does not appear in the assignment list specification or any of its subordinate productions, and is reserved for use by ASN.1 tool developers.

13.2 The "TagDefault" is taken as **`EXPLICIT TAGS`** if it is "empty".

NOTE – Subclause 31.2 gives the meaning of **`EXPLICIT TAGS`**, **`IMPLICIT TAGS`**, and **`AUTOMATIC TAGS`**.

13.3 When the **`AUTOMATIC TAGS`** alternative of "TagDefault" is selected, automatic tagging is said to be selected for the module, otherwise it is said to be not selected. Automatic tagging is a syntactical transformation which is applied (with additional conditions) to the "ComponentTypeLists" and "AlternativeTypeLists" productions occurring within the definition of the module. This transformation is formally specified by 25.8 to 25.10, 27.3 and 29.2 to 29.5 regarding the notations for sequence types, set types and choice types, respectively.

13.4 The **`EXTENSIBILITY IMPLIED`** option is equivalent to the textual insertion of an extension marker ("`...`") in the definition of each type in the module for which it is permitted. The location of the implied extension marker is the last position in the type where an explicitly specified extension marker is allowed. The absence of **`EXTENSIBILITY IMPLIED`** means that extensibility is only provided for those types within the module where an extension marker is explicitly present.

NOTE – **`EXTENSIBILITY IMPLIED`** affects only types. It has no effect on object sets and subtype constraints.

13.5 The "EncodingReferenceDefault" specifies that the "encodingreference" is the default encoding reference for the module. If the "EncodingReferenceDefault" is "empty", then the default encoding reference for the module is **`TAG`**.

NOTE – Annex E contains a list of allowed encoding references, together with the Recommendation | International Standard which specifies the form and meaning of the corresponding encoding instructions.

13.6 The "modulereference" appearing in the "ModuleIdentifier" production is called the module name.

NOTE – The possibility of defining a single ASN.1 module by the use of several occurrences of "ModuleBody" assigned the same "modulereference" was (arguably) permitted in earlier specifications. It is not permitted by this Recommendation | International Standard.

13.7 Module names shall be used only once (except as specified in 13.10) within the sphere of interest of the definition of the module.

13.8 If the "DefinitiveIdentification" is not empty, the denoted object identifier, and any optional "IRIValue", value unambiguously and uniquely identify the same node of the OID tree that identifies the module being defined. No defined value may be used in defining the object identifier value. The "IRIValue" production is specified in 34.3.

NOTE 1 – It is strongly recommended that at least an object identifier value (and preferably an object identifier value plus an OID internationalized resource identifier value) be assigned to the module so that others can unambiguously refer to the module.

NOTE 2 – The question of what changes to a module require a new "DefinitiveIdentification" is not addressed in this Recommendation | International Standard.

13.9 If the "AssignedIdentifier" is not empty, the "ObjectIdentifierValue" and the "DefinedValue" alternatives unambiguously and uniquely identify the module from which reference names are being imported. When the "DefinedValue" alternative of "AssignedIdentifier" is used, it shall be a value of type object identifier. Each "valuereference" which textually appears within an "AssignedIdentifier" shall satisfy one of the following rules:

a) It is defined in the "AssignmentList" of the module being defined, and all "valuereference"s which textually appear on the right side of the assignment statement also satisfy this rule (rule "a") or the next rule (rule "b").

b) It appears as a "Symbol" in a "SymbolsFromModule" whose "AssignedIdentifier" does not textually contain any "valuereference"s.

NOTE 1 – It is recommended that an object identifier be assigned so that others can unambiguously refer to the module.

NOTE 2 – This syntax does not provide for the inclusion of an OID internationalized resource reference (if assigned) to the referenced module, but it is recommended that this be included in an ASN.1 comment.

13.10 The "GlobalModuleReference" in a "SymbolsFromModule" shall appear in the "ModuleDefinition" of another module, except that if it includes a non-empty "DefinitiveIdentification", the "modulereference" may differ in the two cases.

© ISO/IEC 2021 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

NOTE – A different "modulereference" from that used in the other module should only be used when symbols are to be imported from two modules with the same name (the modules being named in disregard of 13.7). The use of alternative distinct names makes these names available for use in the body of the module (see 13.16).

13.11 When both a "modulereference" and a non-empty "AssignedIdentifier" are used in referencing a module, the latter shall be considered definitive.

13.12 When the referenced module has a non-empty "DefinitiveIdentification", the "GlobalModuleReference" referencing that module shall not have an empty "AssignedIdentifier".

13.13 When the "SymbolsExported" alternative of "Exports" is selected:

a) each "Symbol" in "SymbolsExported" shall satisfy one and only one of the following conditions:

i) is only defined in the module being constructed; or

ii) appears exactly once in the "SymbolsImported" alternative of "Imports";

b) every "Symbol" to which reference from outside the module is appropriate shall be included in the "SymbolsExported" and only these "Symbol"s may be referenced from outside the module (subject to the relaxation specified in 13.14); and

c) if there are no such "Symbol"s, then the empty alternative of "SymbolsExported" (not of "Exports") shall be selected.

13.14 When either the "empty" alternative or the **`EXPORTS ALL`** alternative of "Exports" is selected, every "Symbol" defined in the module or imported by the module may be referenced from other modules subject to the restriction specified in 13.13 a).

NOTE – The "empty" alternative of "Exports" is included for backwards compatibility.

13.15 Identifiers that appear in a "NamedNumberList", "Enumeration" or "NamedBitList" are implicitly exported if the typereference that defines them is exported or appears as a component (or subcomponent) within an exported type.

13.16 When the "SymbolsImported" alternative of "Imports" is selected:

a) Each "Symbol" in "SymbolsFromModule" shall either be defined in the module body, or be present in the "Imports" clause, of the module denoted by the "GlobalModuleReference" in "SymbolsFromModule". Importing a "Symbol" present in the "Imports" clause of the referenced module is only allowed if there is only one occurrence of the "Symbol" in that clause, and the "Symbol" is not defined in the referenced module.

NOTE 1 – This does not prohibit the same symbol name defined in two different modules from being imported into another module. However, if the same "Symbol" name appears more than once in the "Imports" clause of module A, that "Symbol" name cannot be exported from A for import to another module B.

b) If the "SymbolsExported" alternative of "Exports" is selected in the definition of the module denoted by the "GlobalModuleReference" in "SymbolsFromModule" the "Symbol" shall appear in its "SymbolsExported".

c) Only those "Symbol"s that appear amongst the "SymbolList" of a "SymbolsFromModule" may appear as the symbol in any "External<X>Reference" which has the "modulereference" denoted by the "GlobalModuleReference" of that "SymbolsFromModule" (where <X> is "Value", "Type", "Object", "Objectclass", or "Objectset").

d) If there are no such "Symbol"s, then the "empty" alternative of "SymbolsImported" shall be selected.

NOTE 2 – An effect of c) and d) is that the statement **`IMPORTS;`** implies that the module cannot contain an "External<X>Reference".

e) All the "SymbolsFromModule" in the "SymbolsFromModuleList" shall include occurrences of "GlobalModuleReference" such that:

i) the "modulereference" in them are all different from each other and from the "modulereference" associated with the referencing module; and

ii) the "AssignedIdentifier", when non-empty, denotes object identifier values which are all different from each other and from the object identifier value (if any) associated with the referencing module.

f) If the "SymbolsFromModule" has a non-empty "SelectionOption", the "AssignedIdentifier" in the "GlobalModuleReference" shall not be empty, and the referenced module shall be determined as follows:

i) If the "SelectionOption" is **`WITH SUCCESSORS`**, the module denoted by the "GlobalModuleReference" is the one that has a DefinitiveIdentification with an object identifier whose last node may be incremented zero or more times. If multiple modules meet this criterion, the denoted module is the one whose object identifier has the last node with the greatest number of increments.

© ISO/IEC 2021 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

ii) If the "SelectionOption" is **WITH DESCENDANTS**, the module denoted by the "GlobalModuleReference" is the one that has a DefinitiveIdentification that identifies the node identified by the "GlobalModuleReference" or one of its subordinates. If multiple modules meet this criterion, the denoted module is the one with the largest object identifier. For this comparison, the arcs are compared successively until one arc is different (selecting the largest arc) or the end of one object identifier is reached (selecting the longer object identifier).

13.17 When the "empty" alternative of "Imports" is selected, the module may still reference "Symbols" defined in other modules by means of an "External<X>Reference".

NOTE – The "empty" alternative of "Imports" is included for backwards compatibility.

13.18 Identifiers that appear in a "NamedNumberList", "Enumeration" or "NamedBitList" are implicitly imported if the typereference that defines them is imported or appears as a component (or subcomponent) within an imported type.

13.19 A "Symbol" in a "SymbolsFromModule" may appear in "ModuleBody" as a "Reference". The meaning associated with the "Symbol" is that which it has in the module denoted by the corresponding "GlobalModuleReference".

13.20 Where the "Symbol" also appears in an "AssignmentList" (deprecated), or appears in one or more other instances of "SymbolsFromModule", it shall only be used in an "External<X>Reference". Where it does not so appear, it shall be used directly as a "Reference".

13.21 The various alternatives for "Assignment" are defined in the following clauses in this Recommendation | International Standard, except as noted otherwise:

Assignment alternative	*Defining subclause*
"TypeAssignment"	16.1
"ValueAssignment"	16.2
"XMLValueAssignment"	16.2
"ValueSetTypeAssignment"	16.6
"ObjectClassAssignment"	Rec. ITU-T X.681 | ISO/IEC 8824-2, 9.1
"ObjectAssignment"	Rec. ITU-T X.681 | ISO/IEC 8824-2, 11.1
"ObjectSetAssignment"	Rec. ITU-T X.681 | ISO/IEC 8824-2, 12.1
"ParameterizedAssignment"	Rec. ITU-T X.683 | ISO/IEC 8824-4, 8.1

The first symbol of every "Assignment" is one of the alternatives of "Reference", denoting the reference name being defined. In no two assignments within an "AssignmentList" shall the reference names be the same.

13.22 "EncodingControlSections" is specified in clause 54.

14 Referencing type and value definitions

14.1 The defined type and value productions:

```
DefinedType ::=
    ExternalTypeReference
|   typereference
|   ParameterizedType
|   ParameterizedValueSetType

DefinedValue ::=
    ExternalValueReference
|   valuereference
|   ParameterizedValue
```

specify the sequences which shall be used to reference type and value definitions. The type identified by a "ParameterizedType" and "ParameterizedValueSetType", and the value identified by a "ParameterizedValue" are specified in Rec. ITU-T X.683 | ISO/IEC 8824-4.

14.2 The "NonParameterizedTypeName" production:

```
NonParameterizedTypeName ::=
    ExternalTypeReference
|   typereference
```

© ISO/IEC 2021 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

| **xmlasn1typename**

is used when an XML tag name is needed to represent an ASN.1 type. If the resulting XML tag name begins with the letters "XML", then a LOW LINE (95) shall be pre-pended to form the "NonParameterizedTypeName".

14.3 The third alternative shall not be used as the "NonParameterizedTypeName" in the "XMLTypedValue" of "XMLValueAssignment" (see 16.2) or of "XMLOpenTypeFieldVal" (see Rec. ITU-T X.681 | ISO/IEC 8824-2, 14.6) when the XML value notation is used in an ASN.1 module if the "xmlasn1typename" is "CHOICE", "ENUMERATED", "SEQUENCE", "SEQUENCE_OF", "SET" or "SET_OF".

NOTE – This restriction is imposed in XML value notation used in an ASN.1 module because these "xmlasn1typename"s do not define an ASN.1 type. The restriction is not present for use of this notation in encoding rules (such as XER, see Rec. ITU-T X.693 | ISO/IEC 8825-4) because XML tag names formed from "xmlasn1typename"s are not used to determine the types that are being encoded.

14.4 Except as specified in 13.19, the "typereference", "valuereference", "ParameterizedType", "ParameterizedValueSetType" or "ParameterizedValue" alternatives shall not be used unless the reference is within the "ModuleBody" in which a type or value is assigned (see 16.1 and 16.2) to the "typereference" or "valuereference".

14.5 The "ExternalTypeReference" and "ExternalValueReference" shall not be used unless the corresponding "typereference" or "valuereference":

a) has been assigned a type or value respectively (see 16.1 and 16.2); or

b) are present in the "Imports" clause,

within the "ModuleBody" used to define the corresponding "modulereference". Referencing a name in the "Imports" clause of another module shall only be allowed if there is no more than one occurrence of the "Symbol" in that clause.

NOTE – This does not prohibit the same "Symbol" defined in two different modules from being imported into another module. However, if the same "Symbol" appears more than once in the **IMPORTS** clause of a module **A**, then that "Symbol" cannot be referenced using module **A** in an external reference.

14.6 An external reference shall be used in a module only to refer to a reference name which is defined in a different module, and is specified by the following productions:

```
ExternalTypeReference ::=
    modulereference
    " . "
    typereference

ExternalValueReference ::=
    modulereference
    " . "
    valuereference
```

NOTE – Additional external reference productions ("ExternalClassReference", "ExternalObjectReference" and "ExternalObjectSetReference") are specified in Rec. ITU-T X.681 | ISO/IEC 8824-2.

14.7 When the referencing module is defined using the "SymbolsImported" alternative of "Imports", the "modulereference" in the external reference shall appear in the "GlobalModuleReference" of exactly one of the "SymbolsFromModule" in the "SymbolsImported". When the referencing module is defined using the "empty" alternative of "Imports", the "modulereference" in the external reference shall appear in the "ModuleDefinition" of the module (different from the referencing module) where the "Reference" is defined.

14.8 Where a "DefinedType" is used as part of notation governed by a "Type" (for example, in a "SubtypeConstraint") then the "DefinedType" shall be compatible with the governing "Type" as specified in clause C.6.2.

14.9 Every occurrence within an ASN.1 specification of a "DefinedValue" is governed by a "Type", and that "DefinedValue" shall reference a value of a type that is compatible with the governing "Type" as specified in clause C.6.2.

15 Notation to support references to ASN.1 components

15.1 There is a requirement for formal reference to components of ASN.1 types, values, etc. for many purposes. One such instance is the need to write text to identify a specific type within some ASN.1 module. This clause defines a notation which can be used to provide such references.

15.2 The notation enables any component of a set or sequence type (which is either mandatorily or optionally present in the type) to be identified.

© ISO/IEC 2021 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

15.3 Any part of any ASN.1 type definition can be referenced by use of the "AbsoluteReference" syntactic construct:

```
AbsoluteReference ::=
    "@" ModuleIdentifier
    "."
    ItemSpec

ItemSpec ::=
    typereference
|   ItemId "." ComponentId

ItemId ::= ItemSpec

ComponentId ::=
    identifier
|   number
|   "*"
```

NOTE – The AbsoluteReference production is not used elsewhere in this Recommendation | International Standard. It is provided for the purposes stated in 15.1.

15.4 The "ModuleIdentifier" identifies an ASN.1 module (see 13.1).

15.5 When the first or second alternative of "DefinitiveIdentification" is used as part of the "ModuleIdentifier", the "DefinitiveIdentification" unambiguously and uniquely identifies the module from which a name is being referenced.

15.6 The "typereference" references any ASN.1 type defined in the module identified by "ModuleIdentifier".

15.7 The "ComponentId" in each "ItemSpec" identifies a component of the type which has been identified by the "ItemId". It shall be the last "ComponentId" if the component it identifies is not a set, sequence, set-of, sequence-of, or choice type.

15.8 The "identifier" form of "ComponentId" can be used if the parent "ItemId" is a set or sequence type, and is required to be one of the "identifier"s of the "NamedType" in the "ComponentTypeLists" of that set or sequence. It can also be used if the "ItemId" identifies a choice type, and is then required to be one of the "identifier"s of a "NamedType" in the "AlternativeTypeLists" of that choice type. It cannot be used in any other circumstance.

15.9 The number form of "ComponentId" can be used only if the "ItemId" is a sequence-of or set-of type. The value of the number identifies the instance of the type in the sequence-of or set-of, with the value "1" identifying the first instance of the type. The value zero identifies a conceptual integer type component (not explicitly present in transfer) that contains a count of the number of instances of the type in the sequence-of or set-of that are present in the value of the enclosing type.

15.10 The "*" form of "ComponentId" can be used only if the "ItemId" is a sequence-of or set-of. Any semantics associated with the use of the "*" form of "ComponentId" apply to all components of the sequence-of and set-of.

NOTE – In the following example:

```
M DEFINITIONS ::= BEGIN
T ::= SEQUENCE {
    a   BOOLEAN,
    b   SET OF INTEGER
}
END
```

the components of "T" could be referenced by text outside an ASN.1 module (or in a comment), such as:

```
-- if (@M.T.b.0 is odd) then:
--        (@M.T.b.* shall be an odd integer)
```

which is used to state that if the number of components in b is odd, all components of b must be odd.

16 Assigning types and values

16.1 A "typereference" shall be assigned a type by the notation specified by the "TypeAssignment" production:

```
TypeAssignment ::=
    typereference
    "::="
    Type
```

The "typereference" shall not be an ASN.1 reserved word (see 12.38).

© ISO/IEC 2021 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

16.2 A "valuereference" shall be assigned a value by the notation specified by either the "ValueAssignment" or "XMLValueAssignment" productions:

```
ValueAssignment ::=
    valuereference
    Type
    "::="
    Value

XMLValueAssignment ::=
    valuereference
    "::="
    XMLTypedValue

XMLTypedValue ::=
    "<" & NonParameterizedTypeName ">"
    XMLValue
    "</" & NonParameterizedTypeName ">"
|   "<" & NonParameterizedTypeName "/>"
```

The value being assigned to the "valuereference" in the "ValueAssignment" is "Value", and is governed by "Type" and shall be a notation for a value of the type defined by "Type" (as specified in 16.3). The value being assigned to the "valuereference" in the "XMLValueAssignment" is "XMLValue" (see 17.7), and shall be a notation for a value of the type defined by "NonParameterizedTypeName" (as specified in 16.4). If this is the "xmlasn1typename" item, then it identifies the ASN.1 built-in type in the corresponding row of Table 4 (see also 14.3). Whitespace is permitted around "XMLValue" in "XMLTypedValue" except where explicitly forbidden (see 41.9 and Rec. ITU-T X.693 | ISO/IEC 8825-4, 31.3.4.1).

16.3 "Value" is a notation for a value of a type as specified in 17.7.

16.4 "XMLValue" is a notation for a value of a type if "XMLValue" is an "XMLBuiltinValue" notation for the type (see 17.10).

16.5 The second alternative of "XMLTypedValue" (use of an XML empty-element tag) can be used only if an instance of the "XMLValue" production is empty.

NOTE – If the "XMLValue" production was an "xmlcstring" containing only white-space, this would not be empty, and the second alternative could not be used.

16.6 A "typereference" can be assigned a value set by the notation specified by the "ValueSetTypeAssignment" production:

```
ValueSetTypeAssignment ::=
    typereference
    Type
    "::="
    ValueSet
```

This notation assigns to "typereference" the type defined as a subtype of the type denoted by "Type" and which contains exactly the values which are specified in or allowed by "ValueSet". The "typereference" shall not be an ASN.1 reserved word (see 12.38), and may be referenced as a type. "ValueSet" is defined in 16.7.

16.7 A value set governed by some type shall be specified by the notation "ValueSet":

```
ValueSet ::= "{" ElementSetSpecs "}"
```

The value set comprises all of the values, of which there shall be at least one, specified by "ElementSetSpecs" (see clause 50).

16.8 The "ValueSetTypeAssignment" production expands into:

```
typereference
Type
"::="
"{" ElementSetSpecs "}"
```

For all purposes, including the application of encoding rules, this is defined to be exactly equivalent to the use of the production:

© ISO/IEC 2021 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

typereference
"::="
Type
"(" ElementSetSpecs ")"

with the same "Type" and "ElementSetSpecs" specifications.

17 Definition of types and values

17.1 A type shall be specified by the notation "Type":

Type ::= BuiltinType | ReferencedType | ConstrainedType

17.2 The built-in types of ASN.1 are specified by the notation "BuiltinType", defined as follows:

BuiltinType ::=
BitStringType
| BooleanType
| CharacterStringType
| ChoiceType
| DateType
| DateTimeType
| DurationType
| EmbeddedPDVType
| EnumeratedType
| ExternalType
| InstanceOfType
| IntegerType
| IRIType
| NullType
| ObjectClassFieldType
| ObjectIdentifierType
| OctetStringType
| RealType
| RelativeIRIType
| RelativeOIDType
| SequenceType
| SequenceOfType
| SetType
| SetOfType
| PrefixedType
| TimeType
| TimeOfDayType

The various "BuiltinType" notations are defined in the following clauses (in this Recommendation | International Standard unless otherwise stated):

BitStringType	22
BooleanType	18
CharacterStringType	40
ChoiceType	29
DateType	38.4.1
DateTimeType	38.4.3
DurationType	38.4.4
EmbeddedPDVType	36
EnumeratedType	20
ExternalType	37
InstanceOfType	Rec. ITU-T X.681 | ISO/IEC 8824-2, Annex C
IntegerType	19
IRIType	34
NullType	24
ObjectClassFieldType	Rec. ITU-T X.681 | ISO/IEC 8824-2, 14.1
ObjectIdentifierType	32
OctetStringType	23

© ISO/IEC 2021 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

RealType	21
RelativeIRIType	35
RelativeOIDType	33
SequenceType	25
SequenceOfType	26
SetType	27
SetOfType	28
PrefixedType	31
TimeType	38.1.1
TimeOfDayType	38.4.2

17.3 The referenced types of ASN.1 are specified by the notation "ReferencedType":

ReferencedType ::=
  DefinedType
| **UsefulType**
| **SelectionType**
| **TypeFromObject**
| **ValueSetFromObjects**

The "ReferencedType" notation provides an alternative means of referring to some other type (and ultimately to a built-in type). The various "ReferencedType" notations, and the way in which the type to which they refer is determined, are specified in the following places in this Recommendation | International Standard unless otherwise stated:

DefinedType	14.1
UsefulType	45.1
SelectionType	30
TypeFromObject	Rec. ITU-T X.681 | ISO/IEC 8824-2, clause 15
ValueSetFromObjects	Rec. ITU-T X.681 | ISO/IEC 8824-2, clause 15

17.4 The "ConstrainedType" is defined in clause 49.

17.5 This Recommendation | International Standard requires the use of the notation "NamedType" in specifying the components of the set types, sequence types and choice types. The notation for "NamedType" is:

NamedType ::= identifier Type

17.6 The "identifier" is used to unambiguously refer to components of a set type, sequence type or choice type in the value notation, in inner subtype constraints and in component relation constraints (see Rec. ITU-T X.682 | ISO/IEC 8824-3). It is not part of the type, and has no effect on the type.

17.7 A value of some type shall be specified by the notation "Value" or by the notation "XMLValue":

Value ::=
  BuiltinValue
| **ReferencedValue**
| **ObjectClassFieldValue**

XMLValue ::=
  XMLBuiltinValue
| **XMLObjectClassFieldValue**

NOTE 1 – "ObjectClassFieldValue" and "XMLObjectClassFieldValue" are defined in Rec. ITU-T X.681 | ISO/IEC 8824-2, 14.6.

NOTE 2 – "XMLValue" is only used in "XMLTypedValue".

17.8 If any part of the "XMLValue" production results in an XML start-tag immediately followed by an XML end-tag, possibly separated by white-space inserted as permitted by 12.1.4 (for example, <field1></field1>), these two XML tags, and any intervening white-space, can be replaced by a single XML empty-element tag (<field1/>).

NOTE – If any white-space character, except white-space inserted as permitted by 12.1.4, is present between the final ">" character of the start tag and the initial "<" character of the end-tag, the condition above is not satisfied.

17.9 Values of the built-in types of ASN.1 can be specified by the notation "XMLBuiltinValue" (see 17.10) or "BuiltinValue", defined as follows:

BuiltinValue ::=
  BitStringValue
| **BooleanValue**
| **CharacterStringValue**
| **ChoiceValue**

© ISO/IEC 2021 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

| **EmbeddedPDVValue**
| **EnumeratedValue**
| **ExternalValue**
| **InstanceOfValue**
| **IntegerValue**
| **IRIValue**
| **NullValue**
| **ObjectIdentifierValue**
| **OctetStringValue**
| **RealValue**
| **RelativeIRIValue**
| **RelativeOIDValue**
| **SequenceValue**
| **SequenceOfValue**
| **SetValue**
| **SetOfValue**
| **PrefixedValue**
| **TimeValue**

Each of the various "BuiltinValue" notations is defined in the same subclause as the corresponding "BuiltinType" notation, as listed in 17.2.

17.10 "XMLBuiltinValue" is defined as follows:

XMLBuiltinValue ::=
XMLBitStringValue
| **XMLBooleanValue**
| **XMLCharacterStringValue**
| **XMLChoiceValue**
| **XMLEmbeddedPDVValue**
| **XMLEnumeratedValue**
| **XMLExternalValue**
| **XMLInstanceOfValue**
| **XMLIntegerValue**
| **XMLIRIValue**
| **XMLNullValue**
| **XMLObjectIdentifierValue**
| **XMLOctetStringValue**
| **XMLRealValue**
| **XMLRelativeIRIValue**
| **XMLRelativeOIDValue**
| **XMLSequenceValue**
| **XMLSequenceOfValue**
| **XMLSetValue**
| **XMLSetOfValue**
| **XMLPrefixedValue**
| **XMLTimeValue**

Each of the various "XMLBuiltinValue" notations is defined in the same clause as the corresponding "BuiltinType" notation, as listed in 17.2 above.

17.11 The referenced values of ASN.1 are specified by the notation "ReferencedValue":

ReferencedValue ::=
DefinedValue
| **ValueFromObject**

The "ReferencedValue" notation provides an alternative means of referring to some other value (and ultimately to a built-in value). The various "ReferencedValue" notations, and the way in which the value to which they refer is determined, are specified in the following places (in this Recommendation | International Standard unless otherwise stated):

DefinedValue	14.1
ValueFromObject	Rec. ITU-T X.681 | ISO/IEC 8824-2, clause 15

17.12 Regardless of whether or not a type is a "BuiltinType", "ReferencedType" or "ConstrainedType", its values can be specified by either a "BuiltinValue" or "ReferencedValue" of that type.

© ISO/IEC 2021 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 08824-1-2021

17.13 The value of a type referenced using the "NamedType" notation shall be defined by the notation "NamedValue", or when used as part of an "XMLValue", by the notation "XMLNamedValue". These productions are:

NamedValue ::= identifier Value

XMLNamedValue ::= "<" & identifier ">" XMLValue "</" & identifier ">"

where the "identifier" is the same as that used in the "NamedType" notation.

NOTE – The "identifier" is part of the notation, it does not form part of the value itself. It is used to unambiguously refer to the components of a set type, sequence type or choice type.

17.14 The implied (see 13.4) or explicit presence of an extension marker (see clause 6) in the definition of a type has no effect on the value notation. That is, the value notation for a type with an extension marker is exactly the same as if the extension marker was absent.

NOTE – Subclause 50.11 prohibits value notation used in a subtype constraint from referencing a value that is not in the extension root of the parent type.

18 Notation for the boolean type

18.1 The boolean type (see 3.8.8) shall be referenced by the notation "BooleanType":

BooleanType ::= `BOOLEAN`

18.2 The tag for types defined by this notation is universal class, number 1.

18.3 The value of a boolean type (see 3.8.85 and 3.8.44) shall be defined by the notation "BooleanValue", or when used as an "XMLValue", by the notation "XMLBooleanValue". These productions are:

BooleanValue ::= `TRUE` | `FALSE`

XMLBooleanValue ::=
EmptyElementBoolean
| TextBoolean

EmptyElementBoolean ::=
"<" & "true" "/>"
| "<" & "false" "/>"

TextBoolean ::=
extended-true
| extended-false

18.4 If an "EmptyElementBoolean" appears in an "XMLValueAssignment", then there shall be no occurrence of "TextBoolean" in that "XMLValueAssignment".

19 Notation for the integer type

19.1 The integer type (see 3.8.48) shall be referenced by the notation "IntegerType":

IntegerType ::=
`INTEGER`
| `INTEGER` "`{`" NamedNumberList "`}`"

NamedNumberList ::=
NamedNumber
| NamedNumberList "," NamedNumber

NamedNumber ::=
identifier "`(`" SignedNumber "`)`"
| identifier "`(`" DefinedValue "`)`"

SignedNumber ::=
number
| "-" number

19.2 The second alternative of "SignedNumber" shall not be used if the "number" is zero.

© ISO/IEC 2021 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 08824-1-2021