Optional items and stylistic advice are grayed-out.
OBJECT IDENTIFIER VALUES

oid1 OBJECT IDENTIFIER ::= {iso standard 2345 modules (0) basic-types (1)}
oid2 OBJECT IDENTIFIER ::= {iso-itu-ts (5)}
oid3 OBJECT IDENTIFIER ::= {oid2 modules(0)}
oid4 OBJECT IDENTIFIER ::= {oid3 basic-types(1)}
oid5 OBJECT IDENTIFIER ::= {oid4

{ 12 840 } US
{ 12 840 113549 } RSA Data Security, Inc.
{ 12 840 113549 1 } RSA Data Security, Inc. PKCS
{ 25 } directory services (X.500)
{ 25 8 } directory services-algorithms

Basic Types

<table>
<thead>
<tr>
<th>Tag</th>
<th>Types</th>
<th>Other Types</th>
<th>Tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean</td>
<td>BOOLEAN</td>
<td>Object Identifier</td>
<td>0[0/1]</td>
</tr>
<tr>
<td>integer</td>
<td>INTEGER</td>
<td>OBJECT IDENTIFIER</td>
<td>0[0/2]</td>
</tr>
<tr>
<td>octet</td>
<td>OCTET STRING</td>
<td>EXTERNAL</td>
<td>0[0/3]</td>
</tr>
<tr>
<td>object1</td>
<td>OBJECT IDENTIFIER</td>
<td>EXTERNAL</td>
<td>0[0/4]</td>
</tr>
<tr>
<td>real</td>
<td>REAL</td>
<td>RELATIVE-OID</td>
<td>0[0/5]</td>
</tr>
<tr>
<td>enum</td>
<td>ENUMERATED</td>
<td>SET</td>
<td>0[10/0]</td>
</tr>
<tr>
<td>sequence</td>
<td>SEQUENCE</td>
<td>OF</td>
<td>0[16/0]</td>
</tr>
<tr>
<td>choice</td>
<td>CHOICE</td>
<td>UTCTime</td>
<td>0[12/0]</td>
</tr>
<tr>
<td>utf8</td>
<td>UTF8_STRING</td>
<td>GeneralizedTime</td>
<td>0[18/0]</td>
</tr>
<tr>
<td>numeric</td>
<td>NUMERIC STRING</td>
<td>Time-OF-DAY</td>
<td>0[32/0]</td>
</tr>
<tr>
<td>ias</td>
<td>IASString</td>
<td>TIME-OF-DAY</td>
<td>0[32/0]</td>
</tr>
<tr>
<td>visible</td>
<td>VisibleString</td>
<td>DATE</td>
<td>0[36/0]</td>
</tr>
<tr>
<td>date</td>
<td>ISO646string</td>
<td>CHARACTER STRING</td>
<td>0[50/0]</td>
</tr>
<tr>
<td>null</td>
<td>IS0646string</td>
<td>Text3Text</td>
<td>0[50/0]</td>
</tr>
</tbody>
</table>

OTHER SYNTAX

EXPORTS ALL

MODULE-NAME.TypeName -- historical alternative to IMPORTS
ABSTRACT-SYNTAX -- rarely used
IMPLICIT TAGS -- historical, but often seen
EXPLICIT TAGS -- historical, but rarely used

SEQUENCE { -- an illustration of tags
    first [0] INTEGER OPTIONAL,
    second [1] EXPLICIT INTEGER,
    last [29] IMPLICIT Userdata
}

APPLICATION 29, [PRIVATE 6] -- More tag illustrations

EXTENSIBILITY IMPLIED -- almost never used

selection < ChoiceType -- occasionally used, not important

PARAMETERS OF SequenceType

-- used to insert common components
SEQUENCE { ...... [29] } -- An exception marker

instance-of

-- Another mechanism for embedding material

oids

-- Complex constraints being applied, and also a Value Set Assignment.
-- My-values IDENTIFIER ::= { ...... } is equivalent to
-- My-values ::= IDENTIFIER { ...... }.

My-values IDENTIFIER ::= {
    {vsel intersection (vselt union vselt)} except vselt4

PrintableString (Size (Namesize)) -- Namesize is defined below
Namesize ::= INTEGER (0..64)

RELATIVE-OID -- A type which carries the tail-end of an object
-- identifier value, with the root statically determined.
-- Sometimes misused to provide an efficient encoding
-- in BER of SEQUENCE OF INTEGER.

EMBEDDED PDV -- Used to embed messages from other specifications,
-- with both the message and the encoding identified
-- at communication time.

EXTERNAL -- Historical (earlier version of EMBEDDED PDV).

PARAMETERIZATION

All assignments defining reference names (type, value, class definitions,
object definitions, object set) can be given a dummy parameter list. Here
we have two dummy parameters -- an INTEGER and Parameter.

Invoke-message (INTEGER: normal-priority, Parameter) ::= SEQUENCE {
    component1 INTEGER DEFAULT normal-priority,
    component2 Parameter }

Now we define our messages as a choice of two possibilities, that differ
only in the default priority and the Type that is to be used:

Messages ::= CHOICE {
    first invoke-message [low-priority, Type1],
    second invoke-message [high-priority, Type2], ...
}

ENCODINGS

Bit-wide PER: A compact binary encoding transferring the
minimum information needed to identify a value.

Byte-wide BER: A type-length-value (TLV) style of encoding

<table>
<thead>
<tr>
<th>T</th>
<th>L</th>
<th>V</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

00-universal 01-application 10-context 11-private
[01..1E] Type tag

DER: An encoding with only one way to encode a
given value, used in security work.

XML:

XER: Encoding ASN.1 values as XML syntax.

There are also Encoding Instructions that can vary XER
and other encodings, for example, to determine which
components of a sequence are to be encoded as XML
attributes.

ECN: An encoding control notation (ECN) is available
to completely determine the encoding of ASN.1 values.

INFORMATION OBJECTS

Use of upper/lower case after "&" is semantically significant.

<MY-SIMPLE-CLASS> ::= <TYPE-IDENTIFIER>

MY-CLASS ::= CLASS {
    &id OBJECT IDENTIFIER  UNIQUE,
    &simple-value ENUMERATED {high, low} DEFAULT low,
    &set-of-values INTEGER OPTIONAL,
    &any-type, &an-infobject SOME-CLASS,
    &a-set-of-objects SOME-OTHER-CLASS
} WITH SYNTAX

-- LITERALS are optional, commas can be used as separators

MY-object MY-CLASS ::= {
    KEY
    ....
    URGENCY high
    VALUE-RANGE { 1..10 | 20..30 }
    PARAMETERS My-type
    SYNTAX defined-syntax
    MATCHING-RULES { at-start | at-end | exact }
}

MY-object-set MY-CLASS ::= {
    object1 | object2 | object3,
    ....
    version2-object
}

Message ::= SEQUENCE {
    -- Has to be OBJECT-ID from the set:
    key MY-CLASS.&id (My-object-set),
    -- Has to be the PARAMETERS for the object with KEY:
    parms MY-CLASS.&any-type (My-object-set) (@key)
}

Variable type value fields and value set fields are out of the scope of this
reference card

OBsolete, not commonly used or deprecated syntax is greyed out below

My

...