File data object specification

Approved Version 1.2.1 – 10 Aug 2007

Open Mobile Alliance

OMA-TS-DS_DataObjFile-V1_2_1-20070810-A

Contents

Tables

1. Scope

The **file** data object is presented in this document. The content-specific aspects of synchronization (filtering keywords, etc…) are listed and clarified.

2. References

2.1 Normative References

[DSDEVDTD] "OMA DS Device Information DTD", Open Mobile Alliance™, "OMA-SUP-DS-DevInf-DTD-V1_2", URL:http://www.openmobilealliance.org

[DSREPU] "SyncML Representation Protocol, Data Synchronization Usage", Open Mobile Alliance™, "OMA-TS-DS_DataSyncRep-V1_2", URL:http://www.openmobilealliance.org

[ISO 8601] ISO 8601, "Data elements and interchange formats- Information interchange--Representation of dates and times", International Organization for Standardization, June, 1988.

[RFC2045] "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", N. Freed & N. Borenstein, November 1966, URL:http://www.ietf.org/rfc/rfc2045.txt

[RFC2119] "Key words for use in RFCs to Indicate Requirement Levels". S. Bradner. March 1997. URL:http://www.ietf.org/rfc/rfc2119.txt

[RFC2234] "Augmented BNF for Syntax Specifications: ABNF", D. Crocker, Ed., P. Overell, November 1997, URL:http://www.ietf.org/rfc/rfc2234.txt

[WBXML] "WAP Binary XML Content Format Specification." WAP Forum. URL:http://www1.wapforum.org/tech/terms.asp?doc=WAP-192-WBXML-20010725-a.pdf

[XML] "Extensible Markup Language (XML) 1.0", World Wide Web Consortium Recommendation, URL:http://www.w3.org/TR/REC-xml

2.2 Informative References

none

3. Terminology and Conventions

3.1 Conventions

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119].

All sections and appendixes, except “Scope” and “Introduction”, are normative, unless they are explicitly indicated to be informative.

3.2 Definitions

Data type The schema used to represent a data object (e.g., text/calendar MIME content type for an iCalendar representation of calendar information or text/directory MIME content type for a vCard representation of contact information).

3.3 Abbreviations

OMA Open Mobile Alliance

4. Introduction

The file data object can be used to represent file which can be stored in various file systems such as FAT, HFS, UFS, etc.

5. XML Usage

The **file** objects are represented in a mark-up language defined by [XML]. The **file** is an XML application. The **file** DTD (Document Type Definition) defines the XML document type used to represent a **file** object. The **file** DTD can be found in Section 9, but it is not necessary to read the DTD in order to understand it.

File objects are specified using well-formed XML. However, the **file** need not be valid XML. That is, the **file** objects do not need to specify the XML declaration or prolog. They only need to specify the body of the XML document. This restriction allows for the **file** objects to be specified with greater terseness than well-formed, valid XML documents.

5.1 Including file object in other XML documents

When using the **file** object as content of another XML document element the mark-up characters MUST be properly escaped or the CDATA sections MUST be used. See [XML]specification for details on character escaping and usage of CDATA sections. See section 10.3 of this document for an example.

5.2 XML Namespaces

File objects to date have no elements that may include elements of the other namespaces.

5.3 XML Attributes

In order to simplify the implementation of the **file** in small devices, the **file** objects have been intentionally designed to use the XML elements only. Currently no XML attributes are being defined for **file** objects.

5.4 WBXML

XML can be viewed as more verbose than alternative binary representations. This is often cited as a reason why it may not be appropriate for low bandwidth network protocols. In most cases, **file** uses shortened element type. This provides a minor reduction in verbosity.

Additionally, the **file** objects can be encoded in a tokenized, binary format defined by [WBXML]. The use of [WBXML] format is external to specification of the **file** and should be transparent to any application. The combination of the use of shortened element type names and an alternative binary format makes **file** competitive, from a compressed format perspective, with alternative, but private, binary representations.

6. MIME Usage

The [RFC2045] Internet standard provides an industry-accepted mechanism for identifying different content types. The **file** object is identified by a MIME media type. The `application/vnd.omads-file+xml` MIME content type MUST be used to indicate the **file** Object wherever such indication is required.

7. Data types

The following basic data type definitions are provided for referencing from other parts of this document.

7.1 datetime

Usage: This value type is used to identify values that specify a precise calendar date and time of day.

Description:

The *datetime* data type is used to identify values that contain a precise calendar date and time of day. The format is based on the [ISO 8601] complete representation, basic format for a calendar date and time of day. The text format is a concatenation of the "date", followed by the LATIN CAPITAL LETTER T character (US-ASCII decimal 84) time designator, followed by the "time" format.

The *datetime* data type expresses time values in two forms:

The form of date and time with UTC offset MUST NOT be used. For example, the following is not valid for a date-time value:

```
...
<datefield>19980119T230000-0800</datefield> <!-- Invalid time format -->
...
```

FORM #1: DATE WITH LOCAL TIME

The date with local time form is simply a date-time value that does not contain the UTC designator nor does it reference a time zone. For example, the following represents January 18, 1998, at 11 PM:

```
...
<datefield>19980118T230000</datefield> <!-- January 18, 1998, 11 PM -->
...
```

This notation of *datetime* type is to be used by devices that have no knowledge of the time zone in which they operate. In this case, the *datetime* value that is being transferred is usually the same as the value that is being stored and shown to the user in the application UI.

FORM #2: DATE WITH UTC TIME

The date with UTC time, or absolute time, is identified by a LATIN CAPITAL LETTER Z suffix character (US-ASCII decimal 90), the UTC designator, appended to the time value. For example, the following represents January 19, 1998, at 0700 UTC:

```
...
<datefield>19980119T070000Z</datefield> <!-- January 19,1998,07:00 UTC -->
...
```

7.2 bool

Usage: To be used for Boolean type fields

Restrictions: A text value that MUST be either "`true`" to indicate Boolean "`true`" or "`false`" to indicate "`false`". If the field is not present its value is assumed to be "`false`".

Example:

```
...
<booleanfield>true</booleanfield> <!-- the field is set to "true" -->
...
```

7.3 text

Usage: To be used for textual fields

Restrictions: If the field is not present its value is assumed to be an empty string.

Example:

```
...
<textfield>Hello World!</textfield>
...
```

7.4 int

Usage: To be used for integer numeric fields.

Restrictions: The format for the integer values is defined here in an ABNF notation [RFC2234].

```
nonzero-digit = "1"/ "2"/ "3"/ "4"/ "5"/ "6"/ "7"/ "8"/ "9"

octal-digit = "0"/ "1"/ "2"/ "3"/ "4"/ "5"/ "6"/ "7"

hexadecimal-digit = "0"/ "1"/ "2"/ "3"/ "4"/ "5"/ "6"/ "7"/ "8"/ "9"
/                   "a"/ "b" / "c"/ "d"/ "e"/ "f"
/                   "A"/ "B"/ "C"/ "D"/ "E"/ "F"

decimal-constant = nonzero-digit *("0" / nonzero-digit)
hexadecimal-constant = ("0x" / "0X") 1*hexadecimal-digit
```

```
octal-constant = "0" 1*octal-digit


integer-value = *1("+" / "-") (decimal-constant / hexadecimal-constant /
octal-constant)
```

Example:

```
...
<negativevalue>-1234</negativevalue>
<positivevalue>1234</positivevalue>
<anotherpositivevalue>+0xfffabc5</anotherpositivevalue>
<octal>010</octal> <!-- octal value equivalent to decimal 8 -->
...
```

8. Mark-up Language Description

8.1 File

Usage: Indicates the beginning of the object

Parent elements: None

Content model:

```
File (name, created?, modified?, accessed?, attributes?, cttype?, body?,
size?, Ext*)
```

8.2 name

Usage: Specifies name of the file.

Parent elements: `File`

Restrictions: This element is mandatory and its value MUST NOT be an empty string The name SHOULD NOT not include any location information (URL, path, etc.).

Content model:

```
name (#PCDATA)
```

8.3 created

Usage: Specifies the date and time when the file was created

Parent elements: `File`

Restrictions: *datetime* type field as specified in section 7.1.

Content model:

```
created (#PCDATA)
```

8.4 modified

Usage: Specifies the date and time when the body of the file object was last changed.

Parent elements: `File`

Restrictions: *datetime* type field as specified in section 7.1.

Content model:

```
modified (#PCDATA)
```

8.5 accessed

Usage: Specifies the date and time when the body of the file object was last accessed.

Parent elements: `File`

Restrictions: *datetime* type field as specified in section 7.1.

Content model:

```
accessed (#PCDATA)
```

8.6 attributes

Usage: Specifies the state of the file system attributes of the file.

Parent elements: `File`

Restrictions:

Content model:

```
attributes (h?, s?, a?, d?, w?, r?, x?)
```

8.6.1 h

Usage: Specifies “hidden” attribute state.

Parent elements: `attributes`

Restrictions: *bool* type field as specified in section 7.2.

Content model:

```
h (#PCDATA)
```

8.6.2 s

Usage: Specifies “system” attribute state.

Parent elements: `attributes`

Restrictions: *bool* type field as specified in section 7.2.

Content model:

```
s (#PCDATA)
```

8.6.3 a

Usage: Specifies “archived” attribute state.

Parent elements: `attributes`

Restrictions: *bool* type field as specified in section 7.2.

Content model:

```
a (#PCDATA)
```

8.6.4 d

Usage: Specifies “delete” attribute state. When the “delete” attribute is set, the file can be deleted.

Parent elements: `attributes`

Restrictions: *bool* type field as specified in section 7.2.

Content model:

```
d (#PCDATA)
```

8.6.5 w

Usage: Specifies "writable" attribute state. If the "writable" attribute is set, the file can be written.

Parent elements: `attributes`

Restrictions: *bool* type field as specified in section 7.2.

Content model:

```
w (#PCDATA)
```

8.6.6 r

Usage: Specifies "readable" attribute state. If the "readable" attribute is set, it is possible to read the contents of the file.

Parent elements: `attributes`

Restrictions: *bool* type field as specified in section 7.2.

Content model:

```
r (#PCDATA)
```

8.6.7 x

Usage: Specifies "executable" attribute state. If the "executable" attribute is set, the contents of the file can be executed by the operating system.

Parent elements: `attributes`

Restrictions: *bool* type field as specified in section 7.2.

Content model:

```
x (#PCDATA)
```

8.7 cttype

Usage: Specifies the content type of the file as defined by [RFC2045].

Parent Elements: `File`

Restrictions:

Content Model:

```
cttype (#PCDATA)s
```

Attributes: None.

Example:

```
<cttype>text/plain</cttype>
```

8.8 body

Usage: contains the file body

Parent elements: `File`

Restrictions: The supporters MUST support this property

Content model:

```
body (#PCDATA)
```

8.8.1 enc

Usage: declares the mechanism used to encode the content of the element. This is used to avoid corrupting the XML content of the element with the presence of characters which do not belong to the valid ranges of characters as defined by the [XML]

Parent elements: `body`

Restrictions: The following table lists standard `enc` values that MUST be understood by the conforming implementations.

Enc	Description
"quoted-printable"	The contents of the element is encoded using quoted-printable algorithm as specified by the section 6.7 of the [RFC2045]
"base64"	The contents of the element is encoded using quoted-printable algorithm as specified by the section 6.8 of the [RFC2045]

Table 1 enc attribute values

If the `enc` attribute is not present, the content is assumed to have no encoding.

The implementations SHOULD NOT use other `enc` attribute values than specified in the [Table 1]. In case of other `enc` values the usage of these encodings MUST conform to the rules defined by the [RFC2045] for `Content-Transfer-Encoding`.

Content model:

```
body enc (CDATA #IMPLIED)
```

8.9 size

Usage: Specifies the size of the file object's body

Parent elements: `File`

Restrictions: *int* type field as specified in section 7.4.

Content model:

```
size (#PCDATA)
```

8.10 Extension fields

8.10.1 Unique naming

If an extension field is required, the following naming convention MUST be followed in order to prevent undesirable field name collisions.

```
x-name        = "x-" vendorid "-" 1*(ALPHA / DIGIT / "-")          ;field name

vendorid      = 3*(ALPHA / DIGIT)                      ;Vendor identification

ALPHA         = %x41-5A / %x61-7A                      ; A-Z / a-z

DIGIT         = %x30-39                                ; 0-9
```

8.10.2 Ext

Usage: Specifies the non-standard, experimental extensions supported by the device. The extensions are specified in terms of the XML element type name and the value.

Parent Elements: `File`

Restrictions: The Ext element type MUST specify the extension element name. It may also specify one or more enumerated values. Multiple non-standard extensions can be specified by specifying the Ext element type multiple times. This element type is optional.

Content Model:

```
Ext (XNam, XVal*)
```

Attributes: None.

Example: The following example specifies a non-standard extension, named "`CliVer`" for a fictitious company, Foo, which takes values of "`5.0`", "`5.01`" or "`5.02`".

```
<Ext>
   <XNam>x-Foo-CliVer</XNam>
   <XVal>5.0</XVal>
   <XVal>5.01</XVal>
   <XVal>5.02</XVal>
</Ext>
```

8.10.3 XNam

Usage: Specifies the name of one of the extension element types.

Parent Elements: `Ext`

Restrictions: The element type is required whenever an `Ext` element is present.

Content Model:

```
XNam (#PCDATA)
```

Attributes: None.

Example:

```
<Ext>
  <XNam>x-Foo-CliVer</XNam>
  <XVal>5.0</XVal>
  <XVal>5.01</XVal>
  <XVal>5.02</XVal>
</Ext>
```

8.10.4 XVal

Usage: Specifies one of the valid values for an extension element type.

Parent Elements: `Ext`

Restrictions:

Content Model:

```
XVal (#PCDATA)
```

Attributes: None.

Example:

```
<Ext>
  <XNam>x-Foo-CliVer</XNam>
  <XVal>5.0</XVal>
  <XVal>5.01</XVal>
  <XVal>5.02</XVal>
</Ext>
```

9. DTD

```
<!--

application/vnd.omads-file+xml V1.2 Document Type Definition


http://www.openmobilealliance.org/tech/DTD/OMA-DS-DataObjFile-DTD-V1_2.dtd


Copyright Open Mobile Alliance Ltd., 2002-2003

          All rights reserved

Terms and conditions of use are available from the

Open Mobile Alliance Ltd. web site at

http://www.openmobilealliance.org/useterms.html -->


<?xml version="1.0" encoding="UTF-8"?>

<!-- Root Element -->

<!ELEMENT File (name, created?, modified?, accessed?, attributes?,
cttype?, body?, size?, Ext*)>


<!—: The supporters MUST support this property. The name does not include
any location information (URL, path, etc.). -->

<!ELEMENT name (#PCDATA)>


<!ELEMENT created (#PCDATA)>

<!ELEMENT modified (#PCDATA)>

<!ELEMENT accessed (#PCDATA)>


<!ELEMENT attributes (h?, s?, a?, d?, w?, r?, x?)>

<!ELEMENT h (#PCDATA)>

<!ELEMENT s (#PCDATA)>

<!ELEMENT a (#PCDATA)>

<!ELEMENT d (#PCDATA)>

<!ELEMENT w (#PCDATA)>

<!ELEMENT r (#PCDATA)>
```

```
<!ELEMENT x (#PCDATA)>


<!ELEMENT cttype (#PCDATA)>


<!ELEMENT body (#PCDATA)>
<!ATTLIST body enc CDATA #IMPLIED>
<!ELEMENT size (#PCDATA)>
<!ELEMENT Ext (XNam, XVal*)>
<!ELEMENT XNam (#PCDATA)>
<!ELEMENT XVal (#PCDATA)>


<!-- End of DTD Definition -->
```

10.SyncML Data Synchronization Usage

The following sections describe the content-specific recommendations for using the data synchronization [DSREPU] protocol with **file** data objects.

10.1 CTCap

Refer to [DSDEVDTD] for further details on the specification of the Device Information DTD.

```
<CTCap>
   <CTType> application/vnd.omads-file+xml</CTType>
   <Property>
      <PropName>name</PropName>
      <DataType>text</DataType>
      <MaxSize>260</MaxSize>
      <DisplayName>File name</DisplayName>
   </Property>
   <Property>
      <PropName>body</PropName>
      <DataType>bin</DataType>
      <MaxSize>65535</MaxSize>
      </NoTruncate>
      <DisplayName>File body</DisplayName>
      <PropParam>
             <ParamName>enc</ParamName>
             <ValEnum>base64</ValEnum>
             <ValEnum>quoted-printable</ValEnum>
      </PropParam>
   </Property>
   <Property>
      <PropName>created</PropName>
      <DataType>datetime</DataType>
      <DisplayName>Date created</DisplayName>
   </Property>
   <Property>
```

```
      <PropName>modified</PropName>
      <DataType>datetime</DataType>
      <DisplayName>Date modified</DisplayName>
   </Property>
   <Property>
      <PropName>accessed</PropName>
      <DataType>datetime</DataType>
      <DisplayName>Date accessed</DisplayName>
   </Property>
   <Property>
      <PropName>cttype</PropName>
      <DataType>text</DataType>
      <DisplayName>File type</DisplayName>
   </Property>
   <Property>
      <PropName>attributes</PropName>
      <DataType>struct</DataType>
      <DisplayName>File Attributes</DisplayName>
      <PropParam>
            <ParamName>h</ParamName>
            <DataType>bool</DataType>
            <DisplayName>Hidden</DisplayName>
      </PropParam>
      <PropParam>
            <ParamName>a</ParamName>
            <DataType>bool</DataType>
            <DisplayName>Archived</DisplayName>
      </PropParam>
   </Property>
</CTCap>
```

10.1.1 enc

The `PropParam` element type with the value of `enc` MUST utilise `ValEnum` element type to indicate the supported encoding algorithms. The example above illustrates a section of `CTCap` that lists the standard encoding algorithms as supported.

10.2 Data Sync Record and Field Level Filtering

10.2.1 File Media Object Filter

Filtering for file objects can be specified using both `Record` and `Field` elements. The set of recommended keywords to support are as follows:

```
ct-filter-keyword = <Any field that is defined for the
application/vnd.omads-file+xml content type in this document except for
the body field>
```

10.3 File object replace example

```
...
<Sync>
...
   <Replace>
      <CmdID>6</CmdID>
      <Meta>
         <Type xmlns='syncml:metinf'>application/vnd.omads-file+xml</Type>
      </Meta>
      <Item>
         <Source>
            <LocURI>123</LocURI>
         </Source>
         ...
         <Data><![CDATA[
            <File>
               <name>OMA-DataObject-File.doc</name>
               <created>20030807T231830</created>
               <modified>20030809T015500</modified>
               <attributes>
```

```
                            <h>false</h>
                            <a>true</a>
                </attributes>
                <body enc="base64"> <--!The content of the file goes here -
->
                ...
                </body>
             </File>]]>
          </Data>
       </Item>
    </Replace>
</Sync>
```

Appendix A Change History (Informative)

A.1 Approved Version History

Reference	Date	Description
OMA-TS-DS_DataObjFile-V1_2-20060710-A	10 Jul 2006	Approved by TP ref#OMA-TP-2006-0239R03-INP_DS_V1_2_for_final_approval
OMA-TS-DS_DataObjFile-V1_2_1-20070122-A	22 Jan 2007	Incorporated CR: OMA-DS-DS_1_2-2006-0006
OMA-TS-DS_DataObjFile-V1_2_1-20070810-A	10 Aug 2007	Prepared for TP notification TP ref # OMA-TP-2007-0326-INP_DS_V1_2_1_ERP_for_Notification