INTERNATIONAL STANDARD

ISO/IEC 14496-22

Fourth edition
2019-01

AMENDMENT 2
2023-01

Information technology — Coding of audio-visual objects —

Part 22:
Open Font Format

AMENDMENT 2: Extending colour font functionality and other updates

Technologies de l'information — Codage des objets audiovisuels —

Partie 22: Format de police de caractères ouvert

AMENDEMENT 2: Extension de la fonctionnalité des polices de couleur et autres mises à jour

Reference number
ISO/IEC 14496-22:2019/Amd. 2:2023(E)

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

Published in Switzerland

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 https://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 29, *Coding of audio, picture, multimedia and hypermedia information*.

A list of all parts in the ISO/IEC 14496 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.

Information technology — Coding of audio-visual objects —

Part 22: **Open Font Format**

AMENDMENT 2: Extending colour font functionality and other updates

4.3

Add the following row to the table defining data types before the row that specifies Offset32:

Offset24	24-bit offset to a table, same as uint24, NULL offset = 0x000000

5.7.1

Replace the entire content of the subclause with the following:

The DSIG table contains the digital signature of the OFF font. Signature formats are widely documented and rely on a key pair architecture. Software developers, or publishers posting material on the Internet, create signatures using a private key. Operating systems or applications authenticate the signature using a public key.

The W3C and major software and operating system developers have specified security standards that describe signature formats, specify secure collections of web objects, and recommend authentication architecture. OFF fonts with signatures will support these standards.

OFF fonts offer many security features:

— Operating systems and browsing applications can identify the source and integrity of font files before using them,

— Font developers can specify embedding restrictions in OFF fonts, and these restrictions cannot be altered in a font signed by the developer.

The enforcement of signatures is an administrative policy that may be supported by the host environment in which fonts are used. Systems may restrict use of unsigned fonts, or may allow policy to be controlled by a system administrator.

Anyone can obtain identity certificates and encryption keys from a certifying agency, such as Verisign or GTE's Cybertrust, free or at a very low cost.

The DSIG table is organized as follows. The first portion of the table is the header.

DSIG Header

Type	Name	Description
uint32	version	Version number of the DSIG table (0x00000001)

Type	Name	Description
uint16	numSignatures	Number of signatures in the table
uint16	flags	Shall be set to 0x0001
SignatureRecord	signatureRecords[numSignatures]	Array of signature records

The version of the DSIG table is expressed as a uint32, beginning at 0. The version of the DSIG table currently used is version 1 (0x00000001).

Permission bit 0 allows a party signing the font to prevent any other parties from also signing the font (counter-signatures). If this bit is set to zero (0) the font may have a signature applied over the existing digital signature(s). A party who wants to ensure that their signature is the last signature can set this bit.

The DSIG header has an array of signature records that specify the format and offset of signature blocks.

SignatureRecord

Type	Name	Description
uint32	format	Format of the signature
uint32	length	Length of signature in bytes
Offset32	signatureBlockOffset	Offset to the signature block from the beginning of the table

Signatures are contained in one or more signature blocks. Signature blocks may have various formats; currently one format is defined. The format identifier specifies both the format of the signature block, as well as the hashing algorithm used to create and authenticate the signature.

Signature Block Format 1

Type	Name	Description
uint16	reserved1	Reserved for future use; set to zero.
uint16	reserved2	Reserved for future use; set to zero.
uint32	signatureLength	Length (in bytes) of the PKCS#7 packet in the signature field.
uint8	signature[signatureLength]	PKCS#7 packet

For more information about PKCS#7 signatures see [10].

For more information about counter-signatures, see [11].

Format 1: For whole fonts, with either TrueType outlines and/or CFF data

PKCS#7 or PKCS#9. The signed content digest is created as follows:

1) If there is an existing DSIG table in the font:

    a) Remove the DSIG table from font.

    b) Remove the DSIG table entry from the Table Directory.

    c) Adjust table offsets as necessary.

    d) Recalculate the checksumAdjustment in the ‘head’ table.

2) Hash the revised font data using a secure one-way hash (such as MD5) to create the content digest.

3) Create the PKCS#7 signature block using the content digest.

4) Create a new DSIG table containing the signature block.

5) Add the DSIG table to the font, adjusting table offsets as necessary.

6) Add a DSIG table entry to the Table Directory.

7) Recalculate the checksumAdjustment in the 'head' table.

Validation of a signature in a font is done by repeating steps 1 – 4 in an in-memory copy of the font file. Note that changing the checksumAdjustment in the last step does not break the signature because verification is done on an in-memory copy with these changes.

Prior to signing a font file, ensure that all the following attributes are true:

— The magic number in the 'head' table is correct.

— Given the numTables value in the Table Directory, the other values in the Table Directory are consistent.

— The table records in the Table Directory are ordered alphabetically by the table tags, and there are no duplicate tags.

— The offset of each table is a multiple of 4. (That is, tables are long word aligned.)

— The first actual table in the file comes immediately after the directory of tables.

— If the tables are sorted by offset, then for all tables i (where index 0 means the table with the smallest offset), Offset[i] + Length[i] <= Offset[i+1] and Offset[i] + Length[i] >= Offset[i+1] - 3. In other words, the tables do not overlap, and there are at most 3 bytes of padding between tables.

— The pad bytes between tables are all zeros.

— The offset of the last table in the file plus its length is not greater than the size of the file.

— The checksums of all tables are correct.

— The 'head' table's checksumAdjustment field is correct.

Signatures for Font Collections

The DSIG table for a Font Collection (TTC) shall be the last table in the TTC file. The offset to the table is put in the TTCHeader (version 2). Signatures of TTC files are expected to be Format 1 signatures.

The signature of a TTC file applies to the entire file, not to the individual fonts contained within the TTC. Signing the TTC file ensures that other contents are not added to the TTC.

Individual fonts included in a font collection should not be individually signed as the process of making the TTC could invalidate the signature on the font.

When DSIG table is created for a collection file, the steps given above are used, with these revisions:

— In step 1: if there is an existing DSIG table referenced in a version 2.0 TTC header, the DSIG table is removed, and the DSIG fields in the header is set to NULL. No recalculation of a checksumAdjustment is required.

— In steps 6 and 7: the DSIG table is added to the file, not to any individual font within the collection. A version 2.0 TTC header is required, with the DSIG fields in the header set to reference the DSIG table.

— Step 8 is not applicable.

See the TTC Header description (subclause 4.6.3) for related information.