Flash ® Media Manifest (F4M) Format Specification

Version 3.0 FINAL

Flash Media Manifest (F4M) Format Specification Version 3.0 FINAL

Adobe Systems Incorporated, 345 Park Avenue, San Jose, California 95110, USA.

Published August 2013

Contents

1 Scope

This specification describes the Adobe® Flash Media Manifest 3.0 file format. The Flash Media Manifest format is used to describe streaming media presentations to media player applications. This specification describes the format and constraints of Flash Media Manifest XML documents, but does not describe in detail the usage of a media manifest with any particular streaming protocol, such as RTMP, RTMFP, or HTTP Dynamic Streaming.

2 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].

“MUST”, “REQUIRED” or “SHALL”, mean that the definition is an absolute requirement of the specification.

“MUST NOT” or “SHALL NOT” means that the definition is an absolute prohibition of the specification.

“SHOULD” or “RECOMMENDED” mean that there may be valid reasons to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.

“SHOULD NOT” or “NOT RECOMMENDED” mean that there may be valid reasons when the particular behavior is acceptable, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.

“MAY” or “OPTIONAL” mean the item is truly optional.

3 Normative References

The following documents contain provisions that, through reference in this text, constitute provisions of this specification.

[FLVSPEC] Video File Format Specification Version 10.1 (http://download.macromedia.com/f4v/video_file_format_spec_v10_1.pdf)

[ISO639-3] ISO 639-3:2007, Codes for the representation of names of languages − Part 3: Alpha-3 code for comprehensive coverage of languages

[RFC2046] IETF RFC 2046, Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types, N. Freed, November 1996 (http://www.ietf.org/rfc/rfc2046.txt?number=2046)

[RFC2119] IETF RFC 2119, Keywords for use in RFCs to Indicate Requirements Levels, S. Bradner, March 1997, (http://www.ietf.org/rfc/rfc2119.txt?number=2119)

[RFC2142] IETF RFC 2142, URN Syntax, R. Moats, May 1997, (http://www.ietf.org/rfc/rfc2142.txt?number=2142)

[RFC3339] IETF RFC 3339, Date and Time on the Internet: Timestamps, G. Klyne, Jully 2002 (http://tools.ietf.org/html/rfc3339).

[RFC6381] IETF RFC 6381, The 'Codecs' and 'Profiles' Parameters for "Bucket" Media Types, R. Gellens, Aug 2011, (http://tools.ietf.org/rfc/rfc6381.txt)

4 Terms and Definitions

F4M – the file extension for a Flash Media Manifest document. Flash Media Manifest files are commonly called "F4M files".

HDS – Adobe® HTTP Dynamic Streaming

Manifest File –a file conforming to the Flash Media Manifest specification, which is used to describe a media presentation.

Multi-Level Manifest –a manifest that is made up of several manifest file documents, which are then composed to form a single info-set. See "set-level manifest" and "stream-level manifest".

Set-Level Manifest –a manifest document that describes only set-level details of a set of content renditions, such as bitrates, screen resolution, language, etc. The document refers to the details of individual streams via a URL reference.

Stream-Level Manifest –a manifest document that describes the details of a single content rendition (as opposed to describing the details of all renditions that make up a presentation).

5 Introduction

5.1 Background

The Flash Media Manifest (F4M) file format is designed to allow digital media publishers to describe complex media presentations to a client device, which could not otherwise be described using a single URL. Using the Flash Media Manifest, it is possible to describe media presentations that contain multiple bitrate renditions, multiple audio language tracks, or sophisticated content protection mechanisms.

While the F4M format was originally introduced to support Adobe HTTP Dynamic Streaming, it is transport protocol agnostic and can be used to describe media presentations for RTMP, RTMFP, or any other media transport scheme.

5.2 What's New in Version 3.0

Version 3.0 of the F4M specification introduces significant new capabilities:

- **Ad cueing information** may be embedded within the manifest via the <cue> and <cueInfo> elements, thereby enabling ad insertion workflows.
- **Video-keyframe-only renditions** may be used to implement trick modes, such as fast forward or rewind.
- **License rotation** enables a single presentation to have multiple encryption keys or policies (i.e. DRM AdditionalHeader's) associated with different time intervals of the presentation.
- **Multiple adaptive sets** may be associated with a given content and type. This can be used to enhance fault tolerance by providing backup hosts for the same content.
- The ability to map **SMPTE timecode**s to stream presentation times has been added to the manifest.
- **Backward compatibility** has been improved via an improved manifest versioning scheme.
- **Best effort fetch** is an optional client behavior that enables enhanced robustness in the face of server-side failures when used with a compatible origin deployment.
- **Alternative audio codec renditions** enable a single presentation to expose audio in multiple codecs, such as both stereo and multi-channel audio.
- The **video codec** of a rendition may be signaled.

See Annex D. "F4M Version History (Informative)" for a full list of changes introduced in F4M 3.0.

6 Manifest Versioning

Manifest files SHALL be versioned via the @version attribute of the root <manifest> element. The format of the version number SHALL be defined as "<major>.<minor>", where both the major and minor fields are integers. The default value of this attribute SHALL be "1.0". This provides some compatibility with the clients and servers that implement version 1.0 of the manifest specification, which did not include an explicit provision for versioning.

Introduced in 2.0: Clients that implement version 2.0 onward SHALL check the @version attribute to ensure the client is capable of interpreting the manifest document. A server SHALL advertise the correct value for @version.

6.1 Compatibility

Introduced in 3.0

For backwards compatibility with future versions, a client or server SHALL ignore any element that it does not understand. In the event that a client or server encounters a known attribute with an unexpected value, the client or server SHALL ignore the offending element.

Two versions of a manifest that share the same major version number SHALL be compatible. If a client or server supports protocol version *x.y*, the client or server MAY process manifests of version *x.z* where *z* may be any minor version number. For example, if a client supports manifest version 2.6, it may safely process manifests of version 2.7, provided it has adhered to the backward compatibility requirements outlined above.

7 Multi-Level Manifest Documents (Informative)

Multi-level manifest documents allow the manifest information required for individual renditions (e.g. HDS bootstrap information) to be provided in a different manifest document than the one carrying set-level information for the presentation. This enables the host managing the set-level metadata to be different from the host managing the stream-level metadata. For example, a set of individual streaming devices could advertise the stream-level metadata for individual renditions of a multi-bitrate adaptive set, while a single content management system (CMS) could advertise set-level metadata that describes the complete adaptive set.

Introduced in 2.0: While the multi-level manifest syntax is composable and could be used to form arbitrary document hierarchies, this specification covers two scenarios to ease implementation and ensure interoperability. The two covered scenarios are:

- **Single Manifest** – A single manifest document is used, with all <media> elements described within the document. This scenario is fully compatible with version 1.0 of the Flash Media Manifest specification.
- **One Set-Level Manifest/Multiple Stream-Level Manifests** – A single set-level manifest document refers to multiple stream-level manifest documents. The set-level document will not contain stream-level details for any <media> element. The stream-level documents will not refer to other manifests.

8 Alternative Content

8.1 Specifying Alternative Content

Introduced in 2.0

It is possible to specify alternative renditions of the content within a presentation. This *alternative content* mechanism can be used to provide multiple language audio tracks for a single video presentation. The alternative content mechanism is distinct from adaptive set behavior. Whereas adaptive set behavior assumes all renditions of a content set are different qualities of the same content, alternative content mechanism enables renditions that specify different actual content to reside in the same presentation.

The alternative content mechanism operates by grouping all renditions according to their content type ("audio", "video", "audio+video", "data", etc.) based on the rendition's @type attribute.

Renditions that are not marked with the @alternate attribute SHALL be considered *primary renditions*, renditions that present the primary content of a given type. The presence of multiple primary renditions SHALL indicate the presence of multiple quality versions of the same primary content.

Renditions marked with the @alternate attribute SHALL be considered *alternative renditions,* renditions that present alternative content of a given type. Alternative renditions SHALL be grouped together based on other distinguishing properties.

Altered in 3.0: The distinguishing properties of alternative audio renditions SHALL be audio language and audio codec, as indicated by the @lang and @audioCodec attributes.

To enable a client to accurately synchronize across multiple renditions, all renditions in the presentation (primary and alternative) SHALL share a common presentation timeline.

8.2 Implementing Multi-Language Audio

A primary use case for alternative content is to allow multiple language audio tracks to be associated with a single presentation. For example, a single presentation may have a primary English language audio rendition and a Spanish language alternative audio rendition.

8.2.1 Manifest Semantics

A manifest MAY advertise audio in multiple languages for a presentation. To do so, the manifest SHALL adhere to the requirements below.

The manifest MAY advertise primary renditions that contain default language audio. Such renditions SHALL be of type "audio" or "video+audio". The language of these renditions SHALL be the default language, as specified by the <lang> element. If the <lang> element is missing, the audio language of all primary renditions SHALL be undetermined.

The manifest MAY advertise primary video content, as specified by a type of "video" or "audio+video".

The manifest MAY advertise alternative audio renditions in an alternative language. Such renditions SHALL be of type "audio" and SHALL specify an audio language via the @lang attribute.

A rendition that contains alternative language audio SHALL be of type “audio” and renditions that contain primary language audio SHALL be of type “audio” or “audio+video” and SHALL either specify their language via the <lang> element or have an undetermined language.

Introduced in 3.0: If multiple alternative audio renditions are present for a given language, renditions that that do not specify an audio codec SHOULD precede those that do specify an audio codec. This ordering enhances backward compatibility with clients that do not support alternative audio codec support.

8.2.2 Selecting the Default Audio Source

A client SHOULD source audio from a primary rendition by default. If no primary rendition contains audio, the client SHOULD source audio from an alternative audio rendition whose language is the same as the default language. If no such rendition exists or the default language is unspecified, the client SHOULD source audio from the first supported alternative audio rendition as it appears in manifest document order.

8.3 Implementing Support for Alternative Audio Codecs

Introduced in 3.0

An additional use case for alternative content is to enable a single presentation to expose audio in multiple codecs, such as both stereo and multi-channel audio.

8.3.1 Manifest Semantics

A manifest MAY advertise audio in multiple audio codecs for a presentation. To do so, the manifest SHALL adhere to the requirements below.

The audio codec of a rendition of type “audio” or “audio+video” SHALL be specified by the @audioCodec attribute. A rendition that is missing the @audioCodec attribute SHALL have an undetermined codec.

The default audio codec SHALL be indicated by the @audioCodec attribute of any primary rendition. All primary renditions SHALL have the same audio codec; That is, if any primary rendition contains the @audioCodec attribute, all primary renditions SHALL have same value for the attribute and if any primary rendition does not contain the @audioCodec attribute, all primary renditions SHALL NOT specify the attribute.

The presence of the @audioCodec attribute on an alternative rendition SHALL indicate a rendition where the specified codec is not the default playback choice.

8.3.2 Client Behavior

When multiple audio renditions are available, a client MAY choose any audio rendition it chooses based on the available language and codec information available to it; there is no requirement specifying the relative selection priorites between audio language and audio codec.

A client MAY adopt playback of a supported codec over an undetermined codec.

A client SHOULD assume same value for the @audioCodec attribute for all primary renditions.

9 Adaptive Sets

An adaptive set is a group of one or more renditions of the same content and same type that may be switched between for the purposes of adaptive bitrate switching. This section defines the semantics used to define the adaptive sets for a given presentation.

In F4M 2.0 and earlier, grouping of renditions into adaptive sets was always implicit; all renditions of the same content and media type were implicitly considered to be in the same adaptive set.

As of version 3.0, it is possible to both implicitly and explicitly group renditions into adaptive sets. This mechanism can be used to enhance resiliency by enabling the specification of backup adaptive sets for content.

All renditions within an adaptive set SHALL have the same *content* and same type. For the duration of this section, content refers to the grouping of renditions by distinguishing properties as specified in section 8. “Alternative Content".

9.1 Implicit Adaptive Sets

Clarified in 3.0, but backward compatible with 2.0

The <media> elements that are directly parented by the <manifest> element SHALL form adaptive sets based on type and content; any pair of <media> elements that are directly parented by the <manifest> element, share the same distinguishing attributes, and are of the same type SHALL be members of the same adaptive set. The adaptive sets formed in this way are known as *implicit adaptive sets*.

A manifest MAY have zero or more implicit adaptive sets for a given content and type.

9.2 Explicit Adaptive Sets

Introduced in 3.0

All <media> elements that are parented by the same <adaptiveSet> element SHALL be members of the same adaptive set. The adaptive sets formed in this way are known as *explicit adaptive sets*.

A manifest MAY have zero or more explicit adaptive sets for a given content and type.

9.3 Adaptive Set Timeline (informative)

Introduced in 3.0

All renditions SHALL share a common presentation timeline. Consequently, all renditions, regardless of which adaptive set they belong to, will share this common timeline.

10 License Rotation (Informative)

Introduced in 3.0

The F4M format enables the usage of *license rotation*, a feature where a single presentation may require different licenses for different time intervals of the presentation. This means that while a particular DRM metadata may be associated with one time interval of a presentation, a different DRM metadata may be associated with a different time interval of the presentation. License rotation provides the basis for implementing features such as enhanced content protection or blackout.

A manifest may optionally enable license rotation. To do so, the manifest should include one or more <drmAdditionalHeaderSet> elements. Each <drmAdditionalHeaderSet> element should be the parent for one or more <drmAdditionalHeader> elements, thereby grouping multiple DRM metadata's into a set. The <media> elements for renditions where license rotation is desired should be associated with <drmAdditonalHeaderSet> elements rather than <drmAdditonalHeader> elements. Finally, the <drmAdditionalHeader> element should include the @startTimestamp attribute which indicates the timestamp of the first sample from where the <drmAdditionalHeader> will apply. The values of the @startTimestamp and @prefetchDeadline attributes should be identical across all renditions of an adaptive set.

See section 10.5. "<drmAdditionalHeaderSet>".

11 XML Document Structure

11.1 <adaptiveSet>

Introduced in 3.0

The <adaptiveSet> element SHALL group together multiple renditions into an explicit adaptive set.

The parent of the this element SHALL be the <manifest> element.

This element SHALL NOT appear in the stream-level manifest of a multi-level manifest document.

The element SHALL contain one or more <media> elements. Each <media> element within a single <adaptiveSet> element SHALL represent the same content and SHALL be the same type of representation. The <baseURL> element will apply to the URLs specified within the <adaptiveSet>.

See section 9. “Adaptive Sets”.

Attributes:

This element MAY have the **alternate, audioCodec, label, lang,** and **type** attributes. When applied to an <adaptiveSet> element, all renditions of the adaptive set SHALL be interpreted as if the attribute was present on their respective <media> elements. For example, if an adaptive set has a @type of “video”, all renditions specified by the child <media> elements shall be assumed to be video renditions. It is not possible to specify conflicting values for the same attribute on a <media> element and its parent <adaptiveSet>, since as specified in section 11.15. “<media>”, a <media> element that is parented by an <adaptiveSet> element SHALL not specify these attributes. Constraints between these attributes SHALL apply to their usage on <adaptiveSet> elements, in the same way that these constraints apply to their usage on a <media> element.

11.2 <baseURL>

The <baseURL> element SHALL be the base URL for all relative (HTTP-based) URLs in the manifest.

The parent of this element SHALL be a <manifest> element.

Altered in 3.0: There SHALL be at most one <baseURL> element per <manifest> element.

This element MAY be present. When present, its value SHALL be prepended to all relative URLs. Relative URLs are URLs that don't begin with "http://" or "https://". Examples of such URLs include the URLs of <media>, <dvrInfo>, <bootstrapInfo>, and <drmAdditionalHeader> elements.)

The scope of the <baseURL> element SHALL be the file it resides in. When using a multi-level manifest document, the relative URLs present in a document (e.g. a set-level or stream-level manifest document) SHALL be relative to the <baseURL> of that document. If <baseURL> is not present, said URLs should be relative to the location of the containing document.

11.3 <bestEffortFetchInfo>

Introduced in 3.0

The <bestEffortFetchInfo> element SHALL contain information needed to enable best-effort fetch support on HTTP streamed media.

The parent of this element SHALL be a <manifest> element.

Multiple instances of this element MAY be present.

The element MAY be present in a set-level manifest. The element SHALL NOT be present in a stream-level manifest.

All renditions within an an adaptive set SHALL be associated with the same <bestEffortFetchInfo> element.

Attributes:

id: SHALL be the ID for this <bestEffortFetchInfo> element. This attribute MAY be present.

If this attribute is missing, the <bestEffortFetchInfo> element SHALL apply to all <media> elements contained within the current document.

If this attribute is present, the <bestEffortFetchInfo> element SHALL apply only to those <media> elements that reference the <bestEffortFetchInfo> element via the @bestEffortFetchInfoId attribute.

If multiple <bestEffortFetchInfo> elements are present within a document, all <bestEffortFetchInfo> elements SHALL specify an @id.

Deprecated attributes:

The following attributes were present in early versions of F4M 3.0. Their usage SHOULD be avoided, except when used to provide backward compatibility.

fragmentDuration: SHALL be the ideal fragment duration as it applies to best effort fetch support. The units of the value SHALL be a decimal number of seconds. The attribute MAY be present. A missing value SHALL indicate an undetermined ideal fragment duration. Usage of this attribute SHOULD be avoided in favor of using the @fragmentDuration attribute of the <bootstrapInfo> element.

segmentDuration: SHALL be the ideal segment duration as it applies to best effort fetch support. The units of the value SHALL be a decimal number of seconds. A value of "0", SHALL indicate an infinite ideal segment duration . The attribute MAY be present. A missing value SHALL indicate an undetermined ideal fragment duration. Usage of this attribute SHOULD be avoided in favor of using the @segmentDuration attribute of the <bootstrapInfo> element.

11.4 <bootstrapInfo>

The <bootstrapInfo> element SHALL be the information needed to perform playback of an HTTP media stream.

The parent of this element SHALL be a <manifest> element.

This element MAY be present. When present, this element SHALL be empty or contain a BASE64 encoded representation of the stream's bootstrap information. When the element is empty, it SHALL have an @url attribute that references the bootstrap information. Either the @url attribute or the inline BASE64 bootstrap info (but not both) SHALL be specified. The bootstrap information SHALL be formed as an 'abst' box as described in [FLVSPEC].

The <bootstrapInfo> element SHALL NOT be present in a set-level manifest.

This <bootstrapInfo> element's scope SHALL be the document it resides in; a stream-level manifest's media SHALL NOT reference a <bootstrapInfo> element from a different stream-level manifest. This ensures there will be no conflicts when two <bootstrapInfo> elements from different documents share the same ID.

Attributes:

fragmentDuration *(introduced in 3.0)*: SHALL be the *ideal fragment duration* for this rendition, expressed as a decimal number of seconds. A precise definition of *ideal fragment duration* is out of scope of this document. This attribute MAY be present. If this attribute is missing, the ideal fragment duration for the rendition is undetermined.

id: SHALL be the ID of this <bootstrapInfo> element. This attribute MAY be present.

If this attribute is not present, the <bootstrapInfo> element SHALL apply to all <media> elements that do not specify the @bootstrapInfoId attribute.

If this attribute is present, the <bootstrapInfo> element SHALL apply only to those <media> elements that have reference the <bootstrapInfo> element via the @bootstrapInfoId attribute.

profile: SHALL be the profile, or type of bootstrapping represented by this element. For the Named Access profile, the value SHALL be "named". For other bootstrapping profiles, some other string SHALL be used (i.e. the field is extensible). This attribute SHALL be present.

segmentDuration *(introduced in 3.0)*: SHALL be the *ideal segment duration* for this rendition, expressed as a decimal number of seconds. A precise definition of *ideal segment duration* is left to the profile. This attribute MAY be present. A precise definition of *ideal segment duration* is out of scope of this document. If this attribute is missing, the ideal segment duration for the rendition is undetermined.

url: SHALL be the URL of a document containing the bootstrap information. Either the @url attribute or inline BASE64 bootstrap info (but not both) SHALL be specified. If the URL is not fully qualified, it SHALL be interpreted according to the relative URL rules, as specified in section 11.2 “<baseURL>”.

11.5 <cue>

Introduced in 3.0

The <cue> element SHALL convey a *splice*, a sequence of time within the presentation where content may be inserted.

The parent of this element SHALL be a <cueInfo> element.

Attributes:

availNum: SHALL be the index of the avail within the total set of avails for the program content. This attribute MAY be present.

availsExpected: SHALL be the expected total number of avails for the program content. This attribute MAY be present.

duration: SHALL be the splice duration, expressed as a decimal number of seconds. This attribute SHALL be present.

id: SHALL be a the ID of this <cue> element. This attribute SHALL be present.

time: SHALL be the stream presentation time at which point the splice should occur, expressed as a decimal number of seconds. This attribute SHALL be present.

type: SHALL be “spliceOut”. This attribute SHALL be present.

programId: SHALL be an identifier for the program content. This attribute MAY be present.

11.6 <cueInfo>

Introduced in 3.0

The <cueInfo> element SHALL be a container element for a sequence of <cue> elements.

The parent of this element SHALL be a <manifest> element.

This element MAY be present. This element SHALL NOT be present in a set-level manifest. This element's scope SHALL be the document it resides in; a <media> element of a stream-level manifest SHALL NOT reference a <cueInfo> element from a different stream-level manifest.

The <cueInfo> elements across all stream level manifests of a given presentation SHALL be symmetric.

The <cueInfo> element SHALL have zero or more child <cue> elements in ascending order of @time.

Attributes:

id: SHALL be the ID of this <cueInfo> element. This attribute SHALL be present.

11.7 <deliveryType>

The <deliveryType> element SHALL specify the means by which content is delivered to the player.

The parent of this element SHALL be a <manifest> element.

The value SHALL be "streaming" or "progressive". This element MAY be present.

If unspecified, the delivery type SHALL be inferred from the media protocol. For media with an RTMP protocol, the default <deliveryType> value SHALL be "streaming". For media with an HTTP protocol, the default delivery type SHALL be "streaming" which indicates Adobe HTTP Dynamic Streaming (HDS) content.

11.8 <drmAdditionalHeader>

The <drmAdditionalHeader> element SHALL be the DRM AdditionalHeader needed for DRM authentication.

The parent of this element SHALL be a <manifest> element or a <drmAdditionalHeaderSet> element.

This element SHALL contain either:

- a BASE64 encoded representation of the DRM AdditionalHeader AMF object, or
- a URL that references the DRM AdditionalHeader AMF object .

Either the @url attribute or the inline BASE64 header (but not both) SHALL be present. The DRM Additional Header object is an AMF0 encoded "|AdditionalHeader" message, as described in [FLVSPEC]

This element MAY be present.

A <drmAdditionalHeader> element SHALL NOT be present in a set-level manifest.

This element's scope SHALL be the document it resides in; a stream-level manifest's media SHALL NOT reference a <drmAdditionalHeader> element from a different stream-level manifest. This ensures there will be no conflicts when two <drmAdditionalHeader> elements from different documents share the same ID.

Attributes:

id: SHALL be the ID of this <drmAdditionalHeader> element. This attribute MAY be present.

If this attribute is not present, this element SHALL apply to all <media> elements that do not specify the @drmAdditionalHeaderId attribute.

If this attribute is present, the <drmAdditionalHeader> element SHALL apply only to those <media> elements that reference the <drmAdditionalHeader> element via the @drmAdditionalHeaderId attribute.

prefetchDeadline *(introduced in 3.0)*: SHALL be a stream presentation time, expressed as a decimal number of seconds. This attribute MAY be present.

If the <drmAdditionalHeader> element is parented by a <drmAdditionalHeaderSet> element, one or both of the @startTimestamp attribute or the @prefetchDeadline attribute SHALL be present.

A client SHALL fetch the license associated with the <drmAdditionalHeader> element prior to the time specified by this attribute. A client SHOULD fetch the license at a random time between the present time and the time specified by this attribute.

startTimestamp *(introduced in 3.0)*: SHALL be the stream presentation timestamp, in fractional seconds, from where the DRM AdditionalHeader applies. This attribute MAY be present.

If the <drmAdditionalHeader> element is parented by a <drmAdditionalHeaderSet> element, one or both of the @startTimestamp attribute or the @prefetchDeadline attribute SHALL be present.

url: SHALL be a URL to a document containing the raw DRM AdditionalHeader. Either the @url attribute or the inline BASE64 header (but not both) SHALL be present. If the URL is not fully qualified, it SHALL be interpreted according to the relative URL rules, as specified in section 11.2 "<baseURL>".

11.9 <drmAdditionalHeaderSet>

Introduced in 3.0

The <drmAdditionalHeaderSet> SHALL be the parent element for one or more <drmAdditionalHeader> elements. It is used to group multiple <drmAdditionalHeader> elements so they can collectively referenced by a <media> element.

The parent of this element SHALL be a <manifest> element.

This element MAY be present.

This element SHALL NOT be present in a set-level manifest.

This element's scope SHALL be the document it resides in; a stream-level manifest's media SHALL NOT reference a <drmAdditionalHeaderSet> element from a different stream-level manifest. This ensures there will be no conflicts when two < drmAdditionalHeaderSet > elements from different documents share the same ID.

Attributes:

id: SHALL be the ID of this <drmAdditionalHeaderSet> element. This attribute MAY be present.

If this attribute is not present, the <drmAdditionalHeaderSet> element SHALL apply to all <media> elements that do not specify the @drmAdditionalHeaderSetId attribute.

If this attribute is present, the <drmAdditionalHeaderSet> element SHALL apply only to those <media> elements that reference the <drmAdditionalHeaderSet> element via the @drmAdditionalHeaderSetId attribute.

11.10 <duration>

The <duration> element SHALL be the duration of the media. All renditions of a presentation are assumed to have the same duration, hence its placement under the document root.

The parent of this element SHALL be a <manifest> element.

The value of this element SHALL be a decimal number of seconds. This element SHALL be present for VOD streams. This element MAY be present for other streams types.

When this element is present for live or DVR content, its value SHALL be zero.

11.11 <dvrInfo>

The <dvrInfo> element SHALL be the information needed to play network DVR media.

The parent of this element SHALL be a <manifest> element.

This element MAY be present.

A <dvrInfo> element SHALL NOT be present in a stream-level manifest.

Attributes:

offline: SHALL indicate whether the stream is offline, or available for playback. A client SHALL not play back the content when the value is true. This attribute MAY be present. The default value SHALL be false.

url: SHALL be a URL to an externally defined *DVR info document*. This attribute MAY be present. If the URL is not fully qualified, it SHALL be interpreted according to the relative URL rules, as specified in section 11.2 "<baseURL>". If the @url attribute is specified, the client SHOULD periodically refetch the DVR info document at the specified URL. The *DVR info document* SHALL be an XML document whose root element is a <dvrInfo> element, and whose structure is the same as this element, with the exception that the <dvrInfo> element SHALL NOT contain a @url attribute.

windowDuration *(introduced in 2.0)*: SHALL be the amount of data, in seconds, that a client can view previous to the latest time in the live stream. A client SHALL only display content within the specified window. If the value of this attribute is '-1', the client MAY view the whole content. The attribute MAY be present. The default value SHALL be '-1'.

11.12 <label>

The <label> element SHALL be a string representing the default user-friendly description of the media. With the exception of the alternative renditions, all renditions of a given presentation are assumed to share the same description.

The parent of this element SHALL be a <manifest> element.

This element MAY be present.

11.13 <lang>

The <lang> element SHALL be a string representing the default language of the piece of media.

The parent of this element SHALL be a <manifest> element.

With the exception of the alternative renditions, all renditions of a given presentation are assumed to share the same default language. The value for this element SHALL be a language code defined in [ISO639-3]. This element MAY be present.

11.14 <id>

The <id> element SHALL represent a unique identifier for the media.

The parent of this element SHALL be a <manifest> element.

This element MAY be present.

11.15 <manifest>

The root of the manifest document SHALL be the <manifest> element. This element SHALL be present.

Attributes:

profile *(introduced in 2.0):* SHALL be the list of HDS profiles supported for this presentation. This attribute MAY be present. If this attribute is not present and the content described by the manifest is HDS, the profile SHALL be assumed to be "urn://profile.adobe.com/F4F". When more than one profile is supported each profile identifier SHALL be separated by a single space.

version (*introduced in 2.0)*: SHALL be the version of the manifest document. The format of the version number SHALL be "<major>.<minor>", where the major and minor fields are both integer values. The value SHALL be "1.0", "2.0", or (*introduced in 3.0*) "3.0". This attribute MAY be present. The default value SHALL be "1.0".

Early revisions of the F4M 2.0 specification indicated manifest version by specifying an alternative namespace for document elements. While this has been changed in later versions of the specification, some tools may still generate set-level manifest files using the namespace "http://ns.adobe.com/f4m/2.0". A client SHOULD recognize manifest files marked with this namespace, and treat them as if the value of @version was "2.0".

11.16 <media>

The <media> element SHALL be one rendition of the media presentation. Each rendition of the presentation SHALL have a corresponding <media> element. Each presentation SHALL contain at least one <media> element.

The parent of this element SHALL be a <manifest> element or (*introduced in 3.0*) an <adaptiveSet> element.

When using a multi-level manifest, no <media> elements of a stream-level manifest SHALL be defined externally; for all <media> elements in a stream-level manifest document of a multi-level manifest document, the @href attribute SHALL NOT be present and the @url attribute SHALL be present.

When using a multi-level manifest, all <media> elements of a set-level manifest SHALL be defined externally; for all <media> elements in a stream-level manifest document of a multi-level manifest document, the @href attribute SHALL be present and the @url attribute SHALL NOT be present.

No <media> elements of a single-level manifest SHALL be defined externally; for all <media> elements in a single-level manifest document, the @href attribute SHALL NOT be present and the @url attribute SHALL be present.

The following attributes SHALL NOT be present in a set-level manifest: @bootstrapInfoId , @cueInfoId, @drmAdditionalHeaderId, @drmAdditionalHeaderSetId, and @url.

The following attributes SHALL NOT be present in a stream-level manifest: @alternate, @audioCodec, @bitrate, @bestEffortFetchInfoId, @height, @href, @label, @lang @streamId, @type, @videoCodec, and @width.

If a <media> element is enclosed in an <adaptiveSet> element, the element SHALL NOT have the @alternate, @audioCodec, @label, @lang, or @type attributes specified. The specification of these attributes on the enclosing <adaptiveSet> element SHALL apply to all renditions in the adaptive set, as specified in section 11.1 "<adaptiveSet>".

Attributes:

alternate: SHALL indicates that this rendition presents alternative content (as opposed to primary content). This attribute MAY be present. When this attribute is present, its value SHALL be "true".

audioCodec (*introduced in 3.0*): SHALL be the audio codec code associated with the rendition. This attribute MAY be present. The value of this attribute SHALL be an audio codec parameter string as described in RFC 6381 (example MPEG-4 HE-AAC Profile is "mp4a.40.5"). This attribute SHALL only be present for renditions of type "audio" or "audio+video".

bitrate: SHALL be the bitrate of the media file, in kilobits per second. If there is only one rendition in the enclosing (implicit or explicit) adaptive set, this attribute MAY be present. If there is more than one rendition in the enclosing adaptive set, this attribute SHALL be present. For renditions of type "video-keyframe-only", the value of this attribute SHALL represent the bitrate of the source content.

bestEffortFetchInfoId (*introduced in 3.0*): SHALL be the ID of a <bestEffortFetchInfo> element that contains the best effort fetch info for this rendition. This attribute MAY be present.

bootstrapInfoId: SHALL be the ID of a <bootstrapInfo> element that contains the bootstrap info for this media rendition. This attribute MAY be present.

cueInfoId (*introduced in 3.0*): SHALL be the ID of a <cueInfo> element that contains the ad insertion cue info for this media rendition. This attribute MAY be present.

drmAdditionalHeaderId: SHALL be the ID of a <drmAdditionalHeader> element that contains the DRM AdditionalHeader for this media file. This attribute MAY be present.

Introduced in 3.0: This attribute SHALL NOT be present if the @drmAdditionalHeaderSetId attribute is present.

drmAdditionalHeaderSetId (*introduced in 3.0*)**:** SHALL be the ID of a <drmAdditionalHeaderSet> element that contains a set of one or more DRM AdditionalHeader's for this media stream. This attribute MAY be present. This attribute SHALL NOT be present if the @drmAdditionalHeaderId attribute is present.

groupspec: SHALL be the group specifier for multicast media. This attribute MAY be present. Multicast is only supported over the RTMFP protocol, and only for a single (non-MBR) stream. If this attribute is present, the @url attribute SHALL be present, and SHALL contain the connection URL, and the @multicastStreamName attribute SHALL also be present.

height: SHALL be the height of the media content, in pixels. This attribute MAY be present.

href *(introduced in 2.0)*: SHALL be the URL of an external F4M file. This attribute MAY be present.

Within a set-level manifest document, this attribute SHALL indicate that stream-level information for this <media> element is contained in an external, stream-level manifest. If the @url attribute is present, the @href attribute SHALL NOT be present. If the URL is not fully qualified, it SHALL be interpreted according to the relative URL rules, as specified in section 11.2 "<baseURL>".

The stream-level manifest indicated by the @href attribute SHALL only contain one <media> element.

label: SHALL be the text description for alternative track. This attribute MAY be present. If the @alternate attribute is also present, then this attribute SHALL be present.

lang: SHALL be the language code associated with the media. The value of this attribute SHALL be one of the language codes described in [ISO639-3]. This attribute MAY be present. If the @alternate attribute is also present, then this attribute SHALL be present.

multicastStreamName: SHALL be the stream name for multicast media. This attribute MAY be present. Multicast is only supported over RTMFP, and only for a single (non-MBR) stream. If this attribute is present, then the @url attribute SHALL contain the connection URL and the "groupspec" attribute SHALL contain the group specifier for the media.

url: SHALL be the URL of the media file. This attribute MAY be present. If the URL is not fully qualified, it SHALL be interpreted according to the relative URL rules, as specified in section 11.2 "<baseURL>". If the @url attribute is present, the @href attribute SHALL NOT be present

streamId: SHALL be the identifier for media file. This attribute MAY be present.

type: SHALL be the type of media this element represents. This attribute is used to facilitate alternate content use cases. The value for this attribute SHALL be "audio+video", "video", "audio", "data", "text", or (*introduced in 3.0*) "video-keyframe-only". This attribute MAY be present. The default value SHALL be "audio+video".

videoCodec (*introduced in 3.0*): SHALL be the video codec code associated with the media. This attribute MAY be present. The value of this attribute SHALL be a video codec parameter string as described in [RFC6381]. For example, the appropriate value for a rendition encoded with H.264 Baseline Profile Level 3 is "avc1.66.30". This attribute SHALL only be present on renditions of type "video", "video-keyframe-only" or "audio+video".

width: SHALL be the width of the media content, in pixels. This attribute MAY be present.

11.17 <metadata>

The <metadata> element SHALL be the stream metadata (i.e. the metadata that is typically dispatched in the onMetadata event) for a single <media> element. This element MAY be present. The parent of this element SHALL be the corresponding <media> element. The metadata SHALL be formed as a BASE64 encoded representation of the stream metadata as an AMF object, as defined in [FLVSPEC].

11.18 <mimeType>

The <mimeType> element SHALL be the MIME type of the media file. It is assumed that all representations of the media have the same MIME type, hence the parent of this element SHALL be a <manifest> element. The value of this element SHALL conform to [RFC2046]. This element MAY be present.

11.19 <startTime>

The <startTime> element SHALL be the date/time at which the media was first (or will first be) made available. It is assumed that all representations of the media have the same start time, hence the parent of this element SHALL be a <manifest> element. The start time SHALL conform to the "date-time" production described in [RFC3339]. This element MAY be present.

11.20 <smpteTimecode>

Introduced in 3.0

The <smpteTimecode> SHALL represent a single sample that maps a stream presentation time to a SMPTE time code. The parent of this element SHALL be the <smpteTimecodes> element.

Attributes:

smpte: SHALL be the timecode in SMPTE format (i.e. "*hour*:*minute*:*second*:*frame*"). This attribute SHALL be present.

timestamp: SHALL be the stream presentation time, expressed in as a decimal number of seconds. This attribute SHALL be present.

date: SHALL be the date corresponding to the SMPTE timecode in the format "*YYYY-MM-DD*". This attribute MAY be present.

timezone: SHALL be the timezone of the SMPTE timecode in the format "[+/-]*hh*:*mm*". This attribute MAY be present.

11.21 <smpteTimecodes>

Introduced in 3.0

The <smpteTimecodes> element SHALL be a container element for the <smpteTimecode> elements which map stream presentation time to SMPTE time codes. The parent of this element SHALL be a <manifest> element. This element MAY be present in a stream level manifest. This element SHALL NOT be present in a set level manifest.

This element SHALL contain zero or more <smpteTimecode> elements. The <smpteTimecode> elements SHOULD appear in increasing stream presentation time order, as described by the @timestamp attribute. Note that this ordering may differ from the ordering by the @smpte attribute.

In order to prevent unbounded growth of the manifest, <smpteTimecode> elements whose @timestamp refers to a time older than the start of the active content availability window SHOULD NOT be included in the manifest.

A server SHOULD include enough <smpteTimecode> to present a reasonably accurate mapping of stream time to SMPTE time code, but not so many as to cause unbounded growth of the manifest. A server SHOULD NOT usually include the SMPTE time code for every video frame.

11.22 <streamType>

The <streamType> element SHALL be a string representing the way in which the media is delivered. The value SHALL be "live", "recorded", or "liveOrRecorded". It is assumed that all representations of the media have the

same stream type, hence its placement under the document root. This element MAY be present. The default value SHALL be “liveOrRecorded”.

12 Mime Type and File Extension

12.1 Mime Type

The mime type of a Flash Media Manifest document SHOULD be “application/f4m”.

12.2 File Extension

The file extension of a Flash Media Manifest document SHOULD be “f4m”.

Annex A. Manifest Examples (Informative)

A.1 Example of recorded RTMP stream

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns="http://ns.adobe.com/f4m/1.0" version="3.0">
   <id>myvideo</id>
   <duration>253</duration>
   <media url="rtmp://example.com/myvideo"/>
</manifest>
```

A.2 Example of recorded HDS presentation with multiple bitrates and DRM

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns="http://ns.adobe.com/f4m/1.0" version="3.0">
   <id>myvideo</id>
   <duration>253</duration>
   <mimeType>video/x-flv</mimeType>
   <streamType>recorded</streamType>
   <baseURL>http://example.com</baseURL>
   <drmAdditionalHeader
     url="http://mydrmserver.com/mydrmadditionalheader"/>
   <bootstrapInfo profile="named" url="/mybootstrapinfo"
                  fragmentDuration="4"/>
   <media url="/myvideo/low" bitrate="408"  width="640"  height="480"/>
   <media url="/myvideo/med" bitrate="908"  width="800"  height="600"/>
   <media url="/myvideo/hi"  bitrate="1708" width="1920" height="1080"/>
</manifest>
```

A.3 Example of live RTMFP stream

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns="http://ns.adobe.com/f4m/1.0" version="3.0">
   <id>myvideo</id>
   <streamType>live</streamType>
   <media url="rtmfp://example.com/myapp" groupspec="G:XYZXYZXYZ"
          multicastStreamName="mystream"/>
</manifest>
```

A.4 Example of live RTMP stream with DVR

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns="http://ns.adobe.com/f4m/1.0" version="3.0">
   <id>myvideo</id>
   <streamType>live</streamType>
   <dvrInfo windowDuration="1800" offline="false" />
   <media url="rtmpe://example.com/myapp" />
</manifest>
```

A.5 Example of HDS presentation with alternative audio renditions

```
<?xml version="1.0" encoding="utf-8"?>
                <manifest xmlns="http://ns.adobe.com/f4m/1.0" version="3.0">
   <id>myvideo</id>
   <streamType>recorded</streamType>
   <duration>100</duration>
   <label>English</label>
   <lang>eng</lang>
   <mimeType>video/mp4</mimeType>
   <baseURL>http://example.com/</baseURL>
   <bootstrapInfo profile="named" id="boot1" fragmentDuration="4">
      (BASE64 encoding of bootstrap information)
   </bootstrapInfo>
   <bootstrapInfo profile="named" id="boot2" fragmentDuration="4">
      (BASE64 encoding of bootstrap information)
   </bootstrapInfo>
   <bootstrapInfo profile="named" id="boot3" fragmentDuration="4">
      (BASE64 encoding of bootstrap information)
   </bootstrapInfo>
   <media url="myvideo" bitrate="1300" bootstrapInfoId="boot1" />
   <media url="myvideo_audio1" bitrate="192" bootstrapInfoId="boot2"
          type="audio" label="German" lang="deu" alternate="true" />
   <media url="myvideo_audio2" bitrate="192" bootstrapInfoId="boot3"
          type="audio" label="Arabic" lang="ara" alternate="true" />
</manifest>
```

A.6 Example of HDS presentation with multiple bitrates and alternative audio

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns="http://ns.adobe.com/f4m/1.0" version="3.0">
   <id>myvideo</id>
   <label>English</label>
   <lang>en</lang>
   <streamType>recorded</streamType>
   <duration>100</duration>
   <mimeType>video/mp4</mimeType>
   <baseURL>http://example.com/</baseURL>
   <bootstrapInfo profile="named" id="boot1 fragmentDuration="4">
      (BASE64 encoding of bootstrap information)
   </bootstrapInfo>
   <bootstrapInfo profile="named" id="boot2" fragmentDuration="4">
      (BASE64 encoding of bootstrap information)
   </bootstrapInfo>
   <bootstrapInfo profile="named" id="boot3" fragmentDuration="4">
      (BASE64 encoding of bootstrap information)
   </bootstrapInfo>
   <bootstrapInfo profile="named" id="boot4" fragmentDuration="4">
      (BASE64 encoding of bootstrap information)
   </bootstrapInfo>
   <bootstrapInfo profile="named" id="boot5" fragmentDuration="4">
      (BASE64 encoding of bootstrap information)
   </bootstrapInfo>
   <media url="myvideo_250"    bitrate="250"  bootstrapInfoId="boot1" />
   <media url="myvideo_500"    bitrate="500"  bootstrapInfoId="boot1" />
   <media url="myvideo_900"    bitrate="900"  bootstrapInfoId="boot1" />
   <media url="myvideo_1300"   bitrate="1300" bootstrapInfoId="boot2" />
   <media url="myvideo_2100"   bitrate="2100" bootstrapInfoId="boot3" />
   <media url="myvideo_audio1" bitrate="192"  bootstrapInfoId="boot4"
          type="audio" label="Espanol" lang="es" alternate="true" />
   <media url="myvideo_audio2" bitrate="192"  bootstrapInfoId="boot5"
          type="audio" label="Chinese" lang="zh" alternate="true" />
</manifest>
```

A.7 Example of Multi-level Manifest

A.7.1 Set-level Manifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns="http://ns.adobe.com/f4m/1.0" version="3.0">
    <id>myVideo</id>
    <baseURL>http://www.example.com/myvideo/</baseURL>
    <streamType>live</streamType>
    <media href="stream250.f4m" bitrate="250" />
    <media href="stream500.f4m" bitrate="500" />
</manifest>
```

A.7.2 Stream-level Manifest (stream250.f4m)

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns="http://ns.adobe.com/f4m/1.0" version="3.0">
   <id>myStream1</id>
   <baseURL>http://www.example.com/data/</baseURL>
   <bootstrapInfo profile="named" id="boot1" fragmentDuration="4">
      (BASE64 encoding of bootstrap information)
   </bootstrapInfo>
   <media url="stream250" bootstrapInfoId="boot1"/>
</manifest>
```

A.7.3 Stream-level Manifest (stream500.f4m)

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns="http://ns.adobe.com/f4m/1.0" version="3.0">
   <id>myStream1</id>
   <baseURL>http://www.example.com/data/</baseURL>
   <bootstrapInfo profile="named" id="boot1" fragmentDuration="4">
      (BASE64 encoding of bootstrap information)
   </bootstrapInfo>
   <media url="stream500" bootstrapInfoId="boot1"/>
</manifest>
```

A.8 Example of alternative audio codecs

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns="http://ns.adobe.com/f4m/1.0" version="3.0">
   <id>myvideo</id>
   <streamType>recorded</streamType>
   <duration>100</duration>
   <label>English</label>
   <lang>en</lang>
   <mimeType>video/mp4</mimeType>
   <baseURL>http://example.com/</baseURL>
   <bootstrapInfo profile="named" id="boot1 fragmentDuration="4">
      (BASE64 encoding of bootstrap information)
   </bootstrapInfo>
   <bootstrapInfo profile="named" id="boot2" fragmentDuration="4">
      (BASE64 encoding of bootstrap information)
   </bootstrapInfo>
   <bootstrapInfo profile="named" id="boot3" fragmentDuration="4">
      (BASE64 encoding of bootstrap information)
   </bootstrapInfo>
   <media url="myvideo"        bitrate="1300" bootstrapInfoId="boot1" />
   <media url="myvideo_audio1" bitrate="192"  bootstrapInfoId="boot2"
          type="audio" audioCodec="ec-3" alternate="true" />
</manifest>
```

A.9 Example of backups

```
<?xml version="1.0" encoding="utf-8"?>
   <manifest xmlns="http://ns.adobe.com/f4m/1.0" version="3.0">
   <id>myVideo</id>
   <streamType>live</streamType>
   <media href="http://server1.example.com/stream250.f4m" bitrate="250"/>
   <media href="http://server1.example.com/stream500.f4m" bitrate="500"/>
   <adaptiveSet>
     <media href="http://server2.example.com/stream300.f4m" bitrate="300"/>
     <media href="http://server2.example.com/stream600.f4m" bitrate="600"/>
   </adaptiveSet>
   <adaptiveSet>
     <media href="http://server3.example.com/stream250.f4m" bitrate="250"/>
     <media href="http://server3.example.com/stream500.f4m" bitrate="500"/>
   </adaptiveSet>
</manifest>
```

The primary audio+video content will be sourced from http://server1.example.com/stream250.f4m or http://server1.example.com/stream500.f4m.

If the primary audio+video content fails, the first backup will be sourced from http://server2.example.com/stream300.f4m or http://server2.example.com/stream600.f4m.

If that backup fails, the next backup will be sourced from http://server3.example.com/stream200.f4m or http://server3.example.com/stream500.f4m.

A.10 Example of backups and alternate audio

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns="http://ns.adobe.com/f4m/1.0" version="3.0">
   <id>myVideo</id>
   <streamType>live</streamType>
   <media href="http://av1.example.com/stream250.f4m" bitrate="250"/>
   <media href="http://av1.example.com/stream500.f4m" bitrate="500"/>
   <media href="http://audio1.example.com/audio.f4m" type="audio"
          alternate="true" lang="es" label="spanish"/>
   <adaptiveSet>
     <media href="http://av2.example.com/stream300.f4m" bitrate="300"/>
     <media href="http://av2.example.com/stream600.f4m" bitrate="600"/>
   </adaptiveSet>
   <adaptiveSet type="audio" alternate="true" lang="es" label="spanish">
     <media href="http://audio2.example.com/audio.f4m"/>
   </adaptiveSet>
</manifest>
```

The primary audio+video content will be sourced from av1.example.com.

If the primary audio+video content fails, the first backup will be sourced from av2.example.com.

The spanish audio content will be sourced from audio1.example.com.

If the spanish audio content fails, the first backup will be sourced from audio2.example.com.

A.11 Example of backups and <baseURL>

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns=" version="3.0">
   <id>myVideo</id>
   <streamType>live</streamType>
   <baseURL>http://server1.example.com/</baseURL>
   <media href="a.f4m"/>
   <adaptiveSet>
     <media href="b.f4m"/>
   </adaptiveSet>
</manifest>
```

The primary audio+video content will be sourced from http://server1.example.com/a.f4m

If the primary audio+video content fails, the first backup will be sourced from http://server1.example.com/b.f4m

A.12 Example presentation with no implicit adaptiveSets

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns=" version=”3.0”>
   <id>myVideo</id>
   <streamType>live</streamType>
   <adaptiveSet>
     <media href="http://server1.example.com/stream.f4m"/>
   </adaptiveSet>
</manifest>
```

The primary audio+video content will be sourced from http://server1.example.com/stream.f4m.

A.13 Example of SMPTE Timecodes

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns="http://ns.adobe.com/f4m/1.0" version="3.0">
   <id>myVideo</id>
   <mimeType> </mimeType>
   <streamType>live</streamType>
   <media url="livestream"  streamId="livestream"
          bootstrapInfoId="boot1" />
   <bootstrapInfo profile="named" id="boot1 fragmentDuration="4">
     (BASE64 encoding of bootstrap information)
   </bootstrapInfo>
   <smpteTimecodes>
     <smpteTimecode timestamp="61646.816" smpte="19:38:11:8"
                    date="2013-04-01" timezone="-08:00" />
     <smpteTimecode timestamp="65246.879" smpte="20:38:11:23"/>
     <smpteTimecode timestamp="68846.942" smpte="21:38:12:7"/>
   </smpteTimecodes>
</manifest>
```

A.14 Example of Ad Signaling Information

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns="http://ns.adobe.com/f4m/1.0" version="3.0">
   <id>myvideo</id>
   <bootstrapInfo profile="named" id="boot1 fragmentDuration="4">
      (BASE64 encoding of bootstrap information)
   </bootstrapInfo>
   <cueInfo id="mycueInfo">
     <cue type="spliceOut" id="1" time="266.061"
          duration="30.5" programId="2354" availNum="1"
          availsExpected="5"/>
     <cue type="spliceOut" id="2" … />
   </cueInfo>
   <media streamId="mystream" url="mystream"
          bootstrapInfoId="boot1" cueInfoId="mycueInfo" />
</manifest>
```

A.15 Example of recorded HDS stream with “video-keyframe-only” media elements

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns="http://ns.adobe.com/f4m/2.0">
   <media url="stream800kbps"  type="audio+video" bitrate="800"  />
   <media url="stream1200kbps" type="audio+video" bitrate="1200" />
   <media url="stream2200kbps" type="audio+video" bitrate="2200" />
   <media url="KFOnly/stream800kbps"  type="video-keyframe-only"
          bitrate="800" />
   <media url="KFOnly/stream1200kbps" type="video-keyframe-only"
          bitrate="1200" />
   <media url="KFOnly/stream2200kbps" type="video-keyframe-only"
          bitrate="2200" />
 </manifest>
```

Annex B. XML Schema (Informative)

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace=http://ns.adobe.com/f4m/1.0
           elementFormDefault="qualified"
           xmlns="http://ns.adobe.com/f4m/1.0"
           xmlns:f4m="http://ns.adobe.com/f4m/1.0"
           xmlns:xs="http://www.w3.org/2001/XMLSchema">
   <xs:element name="manifest">
     <xs:complexType>
       <xs:choice maxOccurs="unbounded">
         <xs:element name="id"
                     type="xs:string"
                     minOccurs="0" />
         <xs:element name="label"
                     type="xs:string"
                     minOccurs="0" />
         <xs:element name="duration
                     type="xs:decimal"
                     minOccurs="0" />
         <xs:element name="mimeType"
                     type="xs:string"
                     minOccurs="0" />
         <xs:element name="streamType"
                     type="streamType"
                     minOccurs="0" />
         <xs:element name="deliveryType"
                     type="deliveryType"
                     minOccurs="0" />
         <xs:element name="startTime"
                     type="xs:dateTime"
                     minOccurs="0" />
         <xs:element name="lang"
                     type="xs:string"
                     minOccurs="0" />
         <xs:element name="baseURL"
                     type="xs:anyURI"
                     minOccurs="0" />
         <xs:element name="drmAdditionalHeader"
                     type="drmAdditionalHeader"
                     minOccurs="0"
                     maxOccurs="unbounded" />
         <xs:element name="drmAdditionalHeaderSet"
                     type="drmAdditionalHeader"
                     minOccurs="0"
                     maxOccurs="unbounded" />
```

```
      <xs:element name="bootstrapInfo"
                  type="bootstrapInfo"
                  minOccurs="0"
                  maxOccurs="unbounded" />
      <xs:element name="dvrInfo"
                  type="dvrInfo"
                  minOccurs="0" />
      <xs:element name="media"
                  type="media"
                  minOccurs="1"
                  maxOccurs="unbounded" />
      <xs:element name="bestEffortFetchInfo"
                  type="bestEffortFetchInfo"
                  minOccurs="0" />
      <xs:element name="adaptiveSet"
                  type="adaptiveSet"
                  minOccurs="0"
                  maxOccurs="unbounded" />
      <xs:element name="cueInfo"
                  type="cueInfo"
                  minOccurs="0"
                  maxOccurs="unbounded" />
    </xs:choice>
    <xs:attribute name="version"
                  type="xs:string"
                  default="1.0" />
  </xs:complexType>
  <xs:unique name="unique-drmAdditionalHeader-id">
    <xs:selector xpath="f4m:drmAdditionalHeader" />
    <xs:field xpath="@id" />
  </xs:unique>
  <xs:unique name="unique-drmAdditionalHeaderSet-id">
    <xs:selector xpath="f4m:drmAdditionalHeaderSet" />
    <xs:field xpath="@id" />
  </xs:unique>
  <xs:unique name="unique-bootstrapInfo-id">
    <xs:selector xpath="f4m:bootstrapInfo" />
    <xs:field xpath="@id" />
  </xs:unique>
</xs:element>
```

```
<xs:simpleType name="streamType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="live" />
    <xs:enumeration value="recorded "/>
    <xs:enumeration value="liveOrRecorded" />
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="deliveryType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="streaming" />
    <xs:enumeration value="progressive" />
  </xs:restriction>
</xs:simpleType>
<xs:complexType name="drmAdditionalHeader">
  <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute name="id"
                    type="xs:string" />
      <xs:attribute name="url"
                    type="xs:anyURI" />
      <xs:attribute name="startTimestamp"
                    type="xs:decimal" />
      <xs:attribute name="prefetchDeadline"
                    type="xs:decimal" />
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="drmAdditionalHeaderSet">
  <xs:attribute name="id"
                type="xs:string" />
  <xs:element name="drmAdditionalHeader"
              type="drmAdditionalHeader"
              minOccurs="0"
              maxOccurs="unbounded" />
</xs:complexType>
```

```
<xs:complexType name="bootstrapInfo">
  <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute name="id"
                    type="xs:string" />
      <xs:attribute name="profile"
                    type="xs:string"
                    use="required" />
      <xs:attribute name="url"
                    type="xs:anyURI" />
      <xs:attribute name="fragmentDuration"
                    type="xs:decimal" />
      <xs:attribute name="segmentDuration"
                    type="xs:decimal" />
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>

<xs:complexType name="dvrInfo">
  <xs:attribute name="id"
                type="xs:string"/>
  <xs:attribute name="url"
                type="xs:anyURI" />
  <xs:attribute name="beginOffset"
                type="xs:decimal" />
  <xs:attribute name="endOffset"
                type="xs:decimal" />
  <xs:attribute name="windowDuration"
                type="xs:decimal" />
  <xs:attribute name="offline"
                type="xs:boolean" />
</xs:complexType>

<xs:complexType name="bestEffortFetchInfo">
  <xs:attribute name="id" type="xs:string" />
  <xs:attribute name="fragmentDuration"
                type="xs:decimal" />
  <xs:attribute name="segmentDuration"
                type="xs:decimal" />
</xs:complexType>
```

```
<xs:complexType name="cueInfo">
  <xs:attribute name="id" type="xs:string"/>
  <xs:element name="cue">
    <xs:complexType>
      <xs:choice maxOccurs="unbounded">
         <xs:attribute name="id"
                       type="xs:string" />
         <xs:attribute name="type"
                       type="xs:string" />
         <xs:attribute name="time"
                       type="xs:decimal" />
         <xs:attribute name="duration"
                       type="xs:decimal" />
         <xs:attribute name="programId"
                       type="xs:decimal" />
         <xs:attribute name="availNum"
                       type="xs:decimal" />
         <xs:attribute name="availsExpected"
                       type="xs:decimal" />
      </xs:choice>
    </xs:complexType>
  </xs:element>
</xs:complexType>
```

```
<xs:complexType name="media">
  <xs:all>
    <xs:element name="moov"
                type="xs:string"
                minOccurs="0" />
    <xs:element name="metadata"
                type="xs:string"
                minOccurs="0" />
    <xs:element name="xmpMetadata"
                type="xs:string"
                minOccurs="0" />
  </xs:all>
  <xs:attribute name="url"
                type="xs:anyURI" />
  <xs:attribute name="bitrate""
                type="xs:positiveInteger" />
  <xs:attribute name="drmAdditionalHeaderId"
                type="xs:string" />
  <xs:attribute name="drmAdditionalHeaderSetId"
                type="xs:string" />
  <xs:attribute name="bootstrapInfoId"
                type="xs:string" />
  <xs:attribute name="bestEffortFetchInfoId"
                type="xs:string" />
  <xs:attribute name="streamId"
                type="xs:string" />
  <xs:attribute name="width"
                type="xs:decimal"/>
  <xs:attribute name="height"
                type="xs:decimal"/>
  <xs:attribute name="groupspec"
                type="xs:string" />
  <xs:attribute name="multicastStreamName"
                type="xs:string" />
  <xs:attributeGroup ref="mediaContent" />
  <xs:attribute name="href"
                type="xs:anyURI" />
  <xs:attribute name="videoCodec "
                type="xs:string" />
</xs:complexType>
```

```
  <xs:attributeGroup id="mediaContent">
    <xs:attribute name="type"
                  type="mediaType" />
    <xs:attribute name="alternate"
                  type="xs:boolean"
                  default="false" />
   <xs:attribute name="label"
                 type="xs:string" />
   <xs:attribute name="lang"
                 type="xs:string" />
   <xs:attribute name="audioCodec"
                 type="xs:string" />
  </xs:attributeGroup>

  <xs:simpleType name="mediaType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="audio" />
      <xs:enumeration value="audio+video" />
      <xs:enumeration value="data" />
      <xs:enumeration value="text" />
      <xs:enumeration value="video" />
      <xs:enumeration value="video-keyframe-only" />
    </xs:restriction>
  </xs:simpleType>

  <xs:complexType name="smpteTimecodes">
    <xs:sequence>
      <xs:element name="smpteTimecode"
                  minOccurs="1"
                  maxOccurs="unbounded">
        <xs:complexType>
           <xs:attribute name="smpte"
                         type="xs:string" />
           <xs:attribute name="timestamp"
                         type="xs:decimal" />
           <xs:attribute name="date"
                         type="xs:date" />
           <xs:attribute name="timezone"
                         type="xs:timezone" />
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>

</xs:schema>
```

Annex C. Supported Audio and Video Codec Values (Informative)

Type	Description	Codec Parameter
Audio	MPEG-4 AAC	`mp4a.40.2`
Audio	MPEG-4 HE-AAC	`mp4a.40.5`
Audio	AC-3	`ac-3`
Audio	Enhanced AC-3	`ec-3`
Video	H.264 Baseline Profile level 3.0	avc1.66.30
Video	H.264 Main Profile level 3.0	avc1.77.30

Annex D. F4M Version History (Informative)

D.1 Changes in Version 3.0

The following normative changes and significant informative changes were introduced in HDS version 3.0:

1. The version numbering scheme of F4M was clearly defined. See section 6.1 “Compatibility”.

2. Ad cueing information may be embedded within the manifest, thereby enabling ad insertion workflows. The <cue> and <cueInfo> elements were defined. The @cueInfo attribute was added to the <media> element. See section 11.5 “

3. <cue>”, section 11.6 “

4. <cueInfo>”, and section 11.16 “<media>”.

5. Video-keyframe-only renditions may be used to implement trick modes, such as fast forward or rewind. “video-keyframe-only” is now an allowed value of the @type attribute for a <media> element. See section 11.16 “<media>”.

6. Support for renditions that use an alternative codec was added. Manifests that use alternative audio codecs may not be backward compatible with HDS 2.0 manifests. The @audioCodec attribute on the <media> element was added. See section 8.3 Implementing Support for Alternative Audio Codecs” and section 11.16 “<media>”.

7. The concept and semantics of adaptive sets were defined. Prior versions of HDS assumed all renditions were members of the same adaptive set, and thus implicitly grouped all renditions (minus alternative audio) into the same adaptive set. The <adaptiveSet> element was added, and the behavior of child <media> elements of an <adaptiveSet> element was specified. See section 9 “Adaptive Sets”, section 11.1 “<adaptiveSet>”, and section 11.16 “<media>”.

   License Rotation enables a single presentation to have multiple encryption keys (i.e. DRM AdditionalHeader’s) associated with different time intervals of the presentation. The @prefetchDeadline and @startTimestamp attributes were added to the <drmAddtionalHeaderSet>. The @drmAdditionalHeaderSetId attribute was added to the <media> element. Specification of both the @drmAdditionalHeaderId and @drmAdditionalHeaderSetId was disallowed. See section 11.8 “<drmAdditionalHeader>”, section 11.9 “

   <drmAdditionalHeaderSet>”, section 10 “License Rotation (Informative)”, and section 11.16 “<media>”.

8. A mapping of SMPTE timecodes to stream presentation times may be embedded within the manifest. The <smpteTimecodes> and <smpteTimeCode> elements were introduced. See section 11.21 "<smpteTimecodes>" and section 11.20 "<smpteTimecode>".

9. The @videoCodec attribute was introduced on the <media> element. It serves as informative metadata for the client. See section 11.16 "<media>".

10. Multiple <baseURL> elements per manifest are now disallowed.

11. The ideal fragment duration and ideal segment duration may be specified for a rendition via the @fragmentDuration and @segmentDuration attributes of the <bootstrapInfo> element. See section 11.4 "<bootstrapInfo>".

12. Best effort fetch is an optional client behavior that enables enhanced robustness in the face of server-side failures. The <bestEffortFetchInfo> element was added to enable HDS best effort fetch behavior. See section 11.3 "<bestEffortFetchInfo>".

13. Alternative content sections were re-worded for clarity. The usage of the term "alternate" in the text has been replaced by "alternative". Attribute names remain unchanged. The concepts of primary and alternative renditions have been clearly defined. The normative requirements regarding alternative language audio process remain unchanged from F4M 2.0. See section 8 "Alternative Content".

14. The parent elements of F4M elements are now explicitly specified.

15. Conflicting requirements surrounding the required attributes of set-level and stream-level manifests were corrected. See section 11.16 "<media>".

16. Ambiguous usages of the terms "rendition", "presentation", "stream set", and "stream" were corrected.