Timed metadata is the key to adding interactivity to live (and VOD) streams produced with Wowza™ streaming technologies. From source to playback, Wowza enables you to produce broadcasts that use timed metadata to offer dynamic, interactive viewing experiences that engage and delight audiences.

**What is timed metadata?**

*Metadata* is data that provides information about other data. Think of an iTunes library, where every song is a piece of data. Every song also has a name, an artist, an album, a genre, and many other descriptive properties, all of which constitute the song’s metadata.

Streams, too, contain metadata. The metadata travels with the stream as it’s encoded and decoded, and it may be used along the way. Two examples of a stream’s metadata are its codec and its resolution. The codec and resolution are specified at the source and used by a player in adaptive bitrate streaming to select a rendition to play for any given viewer’s device and network conditions.

*Timed* metadata just means that a timecode accompanies the piece of data about the stream. When the stream is encoded, the timed metadata is synchronized to audio and video keyframes. The timed metadata flows through a server to the client where, during playback, the timecode serves as a cue point to invoke some action on the data. Perhaps a subtitle translation appears during a streamed opera, or statistics about a baseball team’s winning record appear when it takes the lead in the ninth inning; maybe bidding for an auction item begins, or a game show host starts fielding answers to a question. Other common types of timed metadata include advertising and telemetry information.

**Timed metadata formats**

Many formats and types of timed metadata are used in streaming workflows. For example, SCTE-35 is used to designate where to place ads in MPEG-TS streams, and onTextData is used to display closed captions. These formats, however, have highly structured implementations and specific use cases. For the broadest and most flexible streaming scenario—a source
stream ingested over RTMP and delivered for playback over HLS and WOWZ over WebSockets with custom, user-defined metadata—two formats are used to carry timed metadata with the stream: AMF and ID3.

About AMF

AMF stands for Action Message Format. It’s a binary format developed by Adobe Systems for exchanging messages between servers. In a Wowza streaming workflow, AMF metadata can be received with an ingested source stream over RTMP or WOWZ. AMF metadata can also be injected directly into a Wowza Streaming Engine™ server application. AMF encapsulates the metadata in a packet whose header defines the message length and type, and whose body the custom metadata and an associated timecode.

If you’re delivering the stream over RTMP for Adobe HDS or Flash playback, or using WOWZ over WebSockets using Wowza Streaming Cloud™ with Ultra Low Latency, you’re all set: you can send the AMF metadata through to the player, where it can be used to trigger an action.

Streaming over HLS, however, requires a second timed metadata format: ID3.

About ID3

ID3, which derives from the phrase “identify an MP3,” is the tagging specification that has become the de facto standard for defining metadata in MP3 audio files. Today it’s also used to carry timed metadata in HLS media streams.

An ID3 tag is a container that consists of one or more frames; each frame includes one or more bits of metadata information. The tag is inserted into every HLS media segment at a given time offset. Once the metadata is inserted into the segment, it stays there, so if a live stream is converted to VOD, the timed metadata becomes part of the VOD asset.

Using timed metadata in Wowza workflows

Wowza Streaming Engine, Wowza Streaming Cloud, and Wowza GoCoder™ SDK broadcast and playback apps, and Wowza Player all support timed metadata. The specific ways each one handles AMF and/or ID3 vary, offering wide-ranging and flexible options for using timed metadata across Wowza products.

Using timed metadata in Wowza Streaming Engine

Wowza Streaming Engine can receive AMF metadata from cameras and encoders that support it. Alternatively, you can inject AMF metadata directly into a Wowza Streaming Engine live
application by creating a custom HTTP provider and using the Wowza Streaming Engine Java API. The `com.wowza.wms.amf` package of classes provide methods for working with AMF metadata, and the `IMediaStream` interface provides access to the stream objects.

Similarly, you can use a custom module and the Wowza Streaming Engine Java API to convert AMF metadata to ID3 tags. You can then send the ID3 tags to an HLS stream target in Wowza Streaming Engine or to a Wowza CDN or custom HLS target in Wowza Streaming Cloud.

If you’re using a third-party camera or encoder, check with the manufacturer to see if it supports AMF metadata and for instructions on how to include AMF metadata in live steams. In order for Wowza Streaming Engine to ingest it, however, the metadata must be wrapped in a top-level AMF data object (`AMFDataObj`).

For instructions on injecting or converting metadata in Wowza Streaming Engine see:

- Inject timed metadata using a Wowza Streaming Engine HTTP provider
- Convert timed metadata from AMF to ID3 using the Wowza Streaming Engine Java API

Although the following articles don’t address metadata directly, they provide instructions on how to send streams that can include metadata from Wowza Streaming Engine to Wowza Streaming Cloud:

- Scaling Wowza Streaming Engine with Wowza CDN
- Send a stream from Wowza Streaming Engine to Wowza Streaming Cloud for transcoding using Wowza APIs
- Send a stream from Wowza Streaming Engine to an ultra low latency stream target in Wowza Streaming Cloud using Wowza APIs

Using timed metadata in Wowza Streaming Cloud

Wowza Streaming Cloud can receive AMF metadata from a source stream that originates from:

- A Wowza Streaming Engine live application,
- A broadcasting app created with Wowza GoCoder SDK, or
- A third-party encoder that supports AMF

Unlike with Wowza Streaming Engine, AMF metadata can’t be directly injected into a Wowza Streaming Cloud transcoder.

In order for Wowza Streaming Cloud to convert AMF metadata to ID3 for HLS playback, it must be included in a top-level AMF data object (`AMFDataObj`). Within that AMF data object,
you can include key:value pairs that include AMF data of additional types, such as string, Boolean, list, or even a nested object. The AMF data object must include two properties: (1) a key called `payload` whose value is a string of data to be converted and (2) a key called `wowzaConverter` whose value is `basic_string`.

Once Wowza Streaming Cloud receives AMF metadata with the source stream, it can move directly from the transcoder to an Adobe HDS stream target, where a player can use it to trigger an action during playback.

Alternatively, you can use the Wowza Streaming Cloud REST API to convert AMF metadata to ID3 tags. When you enable the `convertAMFData` property in the REST API, Wowza Streaming Cloud listens for AMF data events in the source stream, parses the data events, maps the events to ID3 tags, and sends the ID3 tags with output renditions to either Wowza CDN or custom HLS stream targets.

The process is different, however, for Wowza Streaming Cloud with ultra low latency, whose streams bypass the Wowza Streaming Cloud transcoder. Ultra low latency streams flow into origin servers over RTMP, and thus they can ingest AMF metadata with the source stream. The origin server sends the AMF metadata to WOWZ-based ultra low latency targets.

In addition, by default, the origin server automatically enables the `convertAMFData` property, and so it converts AMF metadata to ID3 and sends those tags with ultra low latency HLS backup streams.

Through a slightly different metadata workflow, you can leverage ID3 ProgramDateTime (PDT) tags that Wowza Streaming Cloud automatically injects into HLS streams at the beginning of the chunklist as well as the beginning of each chunk. Unlike AMF data, which is more flexible in where it goes and how it’s used, PDT tags are fixed and represent a date and time.

For instructions on using timed metadata in Wowza Streaming Cloud, see:

- Ingest and convert timed metadata with the Wowza Streaming Cloud REST API
- Use timed metadata with an ultra low latency stream target in Wowza Streaming Cloud
- Use timed metadata in HLS streams with the Wowza Streaming Cloud REST API

If you’re using a third-party camera or encoder, check with the manufacturer to see if it supports AMF metadata and for instructions on how to include AMF metadata in live steams.

Using timed metadata with Wowza GoCoder SDK

On the broadcast side, Wowza GoCoder SDK can inject AMF metadata directly into a live
stream as it’s captured. You use the **WOWZDataMap** class to create a data map that contains the metadata. Then, using a **WOWZDataScope** module or stream, you send the data as an event with the stream when it’s broadcast to Wowza Streaming Engine or Wowza Streaming Cloud, where it can be converted to ID3 as needed.

On the playback side, GoCoder SDK is designed to work with ultra low latency streams from Wowza Streaming Cloud. As a result, GoCoder SDK player apps can receive single bitrate streams over WOWZ, including any AMF metadata attached to the stream. To ingest the metadata, use the **WOWZPlayer** class to register a data sink, and then use a **WOWZDataEvent** to listen for and receive the event, which synchronizes with the frame in the event’s timestamp. The SDK then passes the AMF metadata to an RTMP or WOWZ player, including Wowza Player.

At this time, the GoCoder SDK doesn't pass ID3 tags to native Apple and Android player frameworks, but you can build an HLS player independently that will leverage the timed metadata to invoke an action during playback.

For instructions on injecting timed metadata into a Wowza GoCoder SDK broadcast app, see:

- Send and receive events with Wowza GoCoder SDK for iOS
- Send and receive events with Wowza GoCoder SDK for Android

### Using timed metadata with Wowza Player

Wowza Player can listen for and process ID3 tags and AMF metadata. You can use the Wowza Player JavaScript API to create an event listener on a **WowzaPlayer** instance that listens for events in ID3 tags or AMF data.

For instructions, see [Listen for metadata and trigger an action with the Wowza Player JavaScript API](#).

**Note:** For information on whether and how third-party players such as JW Player support ID3 tags, see the player’s documentation.

### Workflow considerations

One of the primary decisions to make when working with timed metadata is where to inject it into the workflow. Should you inject it with the source stream at the encoder or camera, or inject it directly into Wowza Streaming Engine? Either method works, but you’ll want to ensure that the metadata and video are synchronized as closely as possible.
We recommend that you inject the metadata as close as possible to the source—in the source encoder or camera, if possible. When the encoder injects metadata into the source stream, the metadata is associated with the specific video frame being processed at that moment. The source encoder (including a GoCoder SDK broadcasting app) then takes a few seconds to compress and send the stream and the synchronized metadata to Wowza Streaming Engine or Wowza Streaming Cloud. Despite any latency that may occur as the stream travels to the player, the metadata remains as closely aligned as possible with the video frame. That’s why it’s always best, if possible, to inject the metadata with the source at the encoder.

When you inject AMF metadata directly into a Wowza Streaming Engine stream, the AMF metadata is also associated with the specific video frame being processed at that moment. However, latency has typically occurred between the encoder and Wowza Streaming Engine. As a result, the metadata may be injected before the video frame you want to sync to is processed by Wowza Streaming Engine. This may cause the metadata to appear during playback before the desired video frame.

To prevent directly-ingested metadata from getting ahead of the video, either inject the metadata at the source or delay the direct injection into Wowza Streaming Engine: watch RTMP or WOWZ playback of the stream from Wowza Streaming Engine, and initiate the API call to inject the metadata as soon as you see the frame you want to sync to. (There’s less latency with RTMP and WOWZ playback, so you can see the desired frame before it appears for HLS playback.)

Workflow limitations

Finally, when implementing timed metadata with live streams using Wowza technologies and services, keep in mind these limitations:

- Wowza doesn’t provide back-channel communications to data servers. Communications between Wowza and servers that store and process the metadata must be developed separately.
- An AMF-to-ID3 workflow doesn’t work with MPEG-TS or MPEG-DASH streams. MPEG-DASH uses EventStreams or EMSG boxes, neither of which Wowza supports.