Insert keyframes into a transcoded stream with the Wowza Streaming Engine Java API

This article describes how to use the Wowza Streaming Engine™ Java API to plug into the live stream transcoder pipeline to control when keyframes are inserted into a transcoded stream. Through this API, you can both force additional keyframes and stop keyframes from being inserted.

**Note:** Wowza Streaming Engine media server software version 4.5.0.03 or later is required.

The following is module example code that shows this API in action:
// Add a live stream transcoder action listener so we can plug into the transcoding pipeline
((LiveStreamTranscoder) liveStreamTranscoder).addActionListener(new MyTranscoderActionNotifier(appInstance, liveStreamTranscoder, stream));

class MyTranscoderActionNotifier extends LiveStreamTranscoderActionNotifyBase {
    private IApplicationInstance appInstance = null;
    private ILiveStreamTranscoder liveStreamTranscoder = null;
    private IMediaStream stream = null;

    public MyTranscoderActionNotifier(IApplicationInstance appInstance, ILiveStreamTranscoder liveStreamTranscoder, IMediaStream stream) {
        this.appInstance = appInstance;
        this.liveStreamTranscoder = liveStreamTranscoder;
        this.stream = stream;
    }

    // Called when the transcoding session is setup and ready to start
    public void onInitStop(LiveStreamTranscoder liveStreamTranscoder) {
        getLogger().info(CLASSNAME + "#MyTranscoderActionNotifier.onInitStop" + [appInstance.getContextStr() + "/" + liveStreamTranscoder.getStreamName()]);
        try {
            while (true) {
                TranscoderStream transcoderStream = liveStreamTranscoder.getTranscodingStream();
                if (transcoderStream == null) {
                    break;
                }
                TranscoderSession transcoderSession = liveStreamTranscoder.getTranscodingSession();
                TranscoderSessionVideo transcoderVideoSession = transcoderSession.getSessionVideo();
                List<TranscoderSessionVideoEncode> videoEncodes = transcoderVideoSession.getEncodes();
                for (TranscoderSessionVideoEncode videoEncode : videoEncodes) {
                    getLogger().info(CLASSNAME + "#MyTranscoderActionNotifier.onInitStop" + [appInstance.getContextStr() + "/" + liveStreamTranscoder.getStreamName() + "] Add frame listener: rendition:" + videoEncode.getName());
                    // add a transcoder video frame listener
                    videoEncode.addFrameListener(new MyTranscoderVideoEncoderNotifier(appInstance, liveStreamTranscoder, stream));
                }
            }
        } finally {
            // Close the transcoder session
            liveStreamTranscoder.onTranscoderClose();
        }
    }
}


public MyTranscoderVideoEncoderNotifier(IApplicationInstance appInstance, LiveStreamTranscoder liveStreamTranscoder, IMediaStream stream)
{
    this.appInstance = appInstance;
    this.liveStreamTranscoder = liveStreamTranscoder;
    this.stream = stream;
}

public void onBeforeEncodeFrame(TranscoderSessionVideoEncode sessionVideoEncode, TranscoderStreamDestinationVideo destinationVideo, TranscoderVideoEncodeFrameContext frameContext)
{
    TranscoderDecodedFrame decodedFrame = frameContext.getFrameHolder().getEncoderNextFrame();
    if (decodedFrame != null)
    {
        int frameType = FLVUtils.FLV_PFRAME;
        // every 10 frames we are going to force a keyframe
        if ((extraKeyFrames % 10) == 0)
        {
            frameType = FLVUtils.FLV_KFRAME;
            getLogger().info(CLASSNAME + "#MyTranscoderVideoEncoderNotifier.onBeforeEncodeFrame" + appInstance.getContextStr() + "/" + liveStreamTranscoder.getStreamName() + "(" + destinationVideo.getDestination().getName() + "")" + Insert key frame: frameType[source]:" + FLVUtils.frameTypeToString((int)decodedFrame.getFrameType()) + " timecode:" + decodedFrame.getTimecode());
        }
        frameContext.setFrameType(frameType);
        extraKeyFrames++;
    }
}
Where:

- **onAppStart** - Inserts a live stream transcoder listener that listens for new live stream transcoder sessions.

- **MyTranscoderCreateNotifier.onLiveStreamTranscoderCreate** - Is called when a transcoder session is created and inserts an action listener to monitor for transcoder session events.

- **MyTranscoderActionNotifier.onInitStop** - Is called when the session setup is completed and inserts a video frame listener for each encoding rendition that's being generated.

- **MyTranscoderVideoEncoderNotifier.onBeforeEncodeFrame** - Is called before each frame is sent to the encoder, which provides a chance to control the keyframe insertion.

In the example above, a keyframe is forced every 10 seconds. The `decodedFrame.getFrameType()` returns the frame type of the source stream. In your transcoder template, if you've set `/` to `true`, the `frameContext.getFrameType()` is always set to `FLVUtils.FLV_UFRAME` (undefined) by default. This signals to the encoder that it should set the frame type of the encoded stream to the same frame type as the source stream.

If `/` is set to `false`, then the `frameContext.getFrameType()` is set to either `FLVUtils.FLV_PFRAME` (progressive) or `FLVUtils.FLV_KFRAME` (key) based on the `KeyFrameInterval/Interval` (keyframe interval) setting.
In either case, you can force the frame to be either a key or a progressive frame by calling `frameContext.setFrameType(frameType)` with the appropriate frame type.

To debug the keyframe insertion, add the following custom property:

1. In Wowza Streaming Engine Manager, click the Applications tab, and then click the name of your live application (such as live) in the contents panel.
2. On the live application page Properties tab, click Custom in the Quick Links bar.

   **Note:** Access to the Properties tab is limited to administrators with advanced permissions. For more information, see Manage credentials.

3. In the Custom area, click Edit.
4. Click Add Custom Property, specify the following settings in the Add Custom Property dialog box, and then click Add:
   - Path - Select /Root/Application/Streams.
   - Name - Enter debugKeyFrameTimecodes.
   - Type - Select Boolean.
   - Value - Enter true.
5. Click Save, and then restart the live application to apply the changes.