How to live stream with Wowza GoCoder SDK for Android

This article describes how to start, stop, and monitor a live stream broadcast with Wowza GoCoder™ SDK for Android.

Contents

Start streaming
Stop streaming
Monitor the status of the live stream

Start streaming

Use the `startStreaming` method in the `WowzaGoCoder` class to initiate a live stream.

```java
//Make sure streaming isn't already active
if (!this.goCoder.isStreaming()) {
    // Validate the current broadcast config
    WZStreamingError configValidationError = this.goCoder.getConfig().validateForBroadcast();
    if (configValidationError != null) {
        WZLog.error(configValidationError);
    } else {
        // Start the live stream
        this.goCoder.startStreaming(statusCallback);
    }
}
```

Stop streaming

Use the `endStreaming` method in the `WowzaGoCoder` class to stop a live stream.
Monitor the status of the live stream

Use the `statusCallback` argument for the `startStreaming` and `endStreaming` methods to implement the `WZStatusCallback` interface. This interface defines the following callbacks that are invoked during the lifetime of a live stream broadcast:

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WZState.IDLE</td>
<td>The stream is inactive. This is the status before and after the broadcast.</td>
</tr>
<tr>
<td>WZState.STARTING</td>
<td>The audio and video inputs are initializing and the stream is connecting to a Wowza Streaming Engine server or the Wowza Streaming Cloud service.</td>
</tr>
<tr>
<td>WZState.READY</td>
<td>The audio, video, and stream connection are successfully initialized.</td>
</tr>
<tr>
<td>WZState.RUNNING</td>
<td>The broadcast is actively streaming.</td>
</tr>
<tr>
<td>WZState.STOPPING</td>
<td>The streaming connection is closing. After the connection closes, the broadcast status returns to <code>WZState.IDLE</code>.</td>
</tr>
</tbody>
</table>

The following example demonstrates successful status transitions as well as an error status.

```javascript
//
// Methods implemented for the WZStatusCallback interface
//
```
WZStatusCallback statusCallback = new WZStatusCallback() {
    @Override
    public void onSuccess(final WZStatus goCoderStatus) {
        switch (state) {
            case WZState.IDLE:
                WZLog.info(TAG, "Streaming is not active");
                break;

            case WZState.STARTING:
                WZLog.info(TAG, "Broadcast initialization");
                break;

            case WZState.READY:
                WZLog.info(TAG, "Ready to begin streaming");
                break;

            case WZState.RUNNING:
                WZLog.info(TAG, "Streaming is active");
                break;

            case WZState.STOPPING:
                WZLog.info(TAG, "Broadcast shutdown");
                break;
        }
    }

    // If an error is reported by the GoCoder SDK, display a toast message containing the error details
    @Override
    public void onError(final WZStatus goCoderStatus) {
        // Run this on the UI thread
        new Handler(Looper.getMainLooper()).post(new Runnable() {
            @Override
            public void run() {
                Toast.setText("Live Streaming Error: " + goCoderStatus.getLastErrorMessage().getErrorMessage(), LENGTH_LONG).show();
            }
        });
    }
};
If you're having problems or want to discuss this article, post in our forum.