Send and receive events with Wowza GoCoder SDK for Android

Learn how to send events to a Wowza Streaming Engine™ media server software module and, optionally, to receive responses using the WOWZData family of classes in the Wowza GoCoder™ SDK for Android.

There are three primary ways to use the WOWZData classes. The steps required to implement each vary. The three primary uses are:

- Pass data, such as AMF metadata, in a stream through Wowza Streaming Engine using the RTMP or WOWZ protocol.
- Pass data in a stream through Wowza Streaming Engine to other formats, such as Apple HLS.
- Send events from a Wowza Streaming Engine server to a Wowza GoCoder SDK-based app.

Contents

Create a Wowza Streaming Engine server module
Configure an app to send events and receive responses
Add metadata to outgoing streams
More resources

Create a Wowza Streaming Engine server module

If you’re using a protocol other than WOWZ or RTMP to deliver a stream to the player or you’re sending responses from Wowza Streaming Engine back to a Wowza GoCoder SDK-based app, you must create a module in Wowza Streaming Engine.

The following module example listens to the incoming stream for events named onTextData. If it receives an onTextData event, it reverses the event’s data payload and sends it back to the GoCoder SDK-based app using another onTextData event:
Configure an app to send events and receive responses

Next, add the capability to send events to and receive responses from the Wowza Streaming Engine module to the app.
The following example sends a module-scope event without a callback:

```java
public boolean onTouchEvent(MotionEvent event) {
    if (event.getAction() == MotionEvent.ACTION_DOWN &&
        mWZBroadcast != null && mWZBroadcast.getStatus().isRunning()) {
        WOWZDataMap dataEventParams = new WOWZDataMap();
        dataEventParams.put("x", event.getX());
        dataEventParams.put("y", event.getY());
        dataEventParams.put("occurred", event.getEventTime());
        mWZBroadcast.sendDataEvent(WOWZDataScope.MODULE, "onScreenPress", dataEventParams);
        Toast.makeText(this, "onScreenPress() event sent to server module", Toast.LENGTH_LONG).show();
        return true;
    } else
        return super.onTouchEvent(event);
}
```

The following example sends a module-scope event with a callback:
public void onPing(View v) {
//
// Sending an event to a server module method (with a result callback)
//
if (mWZBroadcast != null && mWZBroadcast.getStatus().isRunning()) {
    mBtnPing.setEnabled(false);

    mWZBroadcast.sendDataEvent(WOWZDataScope.MODULE, "onGetPingTime", new WOWZData
Event.ResultCallback() {
        @Override
        public void onWOWZDataEventResult(final WOWZDataMap resultParams, boolean
isError) {
            if (resultParams != null) {
                final String result = isError ? "Ping attempt failed (" + resultPa
rams.get("code").toString() + ") : "Ping time: " + resultParams.get("pingTime") + "ms
;

                new Handler(Looper.getMainLooper()).post(new Runnable() {
                    @Override
                    public void run() {
                        Toast.makeText(EventActivity.this, result, Toast.L
NGTH).show();
                        mBtnPing.setEnabled(true);
                    }
                });
            }
        }
    });
}
}

The following example receives a response:
The following example sends a stream-scope event without a callback:
public void onConfigurationChanged(Configuration newConfig) {
    super.onConfigurationChanged(newConfig);
    //
    // Sending an event to all stream subscribers
    //
    // If a broadcast is active and the device orientation changes,
    // send a stream data event containing the device orientation
    // and rotation
    if (mWZBroadcast != null && mWZBroadcast.getStatus().isRunning()) {
        WOWZDataMap dataEventParams = new WOWZDataMap();
        dataEventParams.put("deviceOrientation",
                newConfig.orientation == Configuration.ORIENTATION_LANDSCAPE ? "landscape" : "portrait");
        Display display = ((WindowManager)
                getSystemService(Context.WINDOW_SERVICE)).getDefaultDisplay();
        int displayRotation = display.getRotation();
        switch (displayRotation) {
            case Surface.ROTATION_0:
                dataEventParams.put("deviceRotation", 0);
                break;
            case Surface.ROTATION_90:
                dataEventParams.put("deviceRotation", 90);
                break;
            case Surface.ROTATION_180:
                dataEventParams.put("deviceRotation", 180);
                break;
            case Surface.ROTATION_270:
                dataEventParams.put("deviceRotation", 270);
                break;
        }
        mWZBroadcast.sendDataEvent(WOWZDataScope.STREAM, "onDeviceOrientation", dataEventParams);
        Toast.makeText(this, "onDeviceOrientation() event sent to stream subscribers", Toast.LENGTH_LONG).show();
    }
}
More resources

- About timed metadata in Wowza live streaming workflows
- GoCoder SDK reference documentation for Android

If you're having problems or want to discuss this article, post in our forum.