Publishing/playback channel quality control

Since build 1.1.0.20 it is possible to receive messages containing current publishing or playback bitrate from server and to control channel quality based on client and server bitrates difference.

To enable channel quality control, use the method `Stream.enableConnectionQualityCalculation()`

```
publishStream.enableConnectionQualityCalculation(true);
```

Then register a callback function to get quality metric and current client and server bitrate values averaged by Kalman filter

```
publishStream.setConnectionQualityCallback((quality, clientRate, serverRate)
-> {
    updateQualityStatus(quality, mUpdateQualityStatus);
});
```

Usage example:

code of handler registration, code of handler

```
private TextView mUpdateQualityStatus;
...
publishStream.enableConnectionQualityCalculation(true);
publishStream.setConnectionQualityCallback((quality, clientRate, serverRate)
-> {
    updateQualityStatus(quality, mUpdateQualityStatus);
});
...
public void updateQualityStatus(ConnectionQuality quality, TextView textView)
{
    int color;
    switch (quality) {
        case BAD: color = Color.RED; break;
        case GOOD: color = Color.YELLOW; break;
        case PERFECT: color = Color.GREEN; break;
        case UPDATE:
        case UNKNOWN:
        default: color = Color.LTGRAY;
    }
    runOnUiThread(() -> {
        textView.setText(quality.toString());
        textView.setTextColor(color);
    });
}
```

This feature can be used to detect if video is muted in playing stream. In this case, current server bitrate should be passed to handler:

```
playStream.enableConnectionQualityCalculation(true);
playStream.setConnectionQualityCallback((quality, clientRate, serverRate) ->
{
    checkVideoMuted(quality, clientRate, serverRate);
});
...
public void checkVideoMuted(ConnectionQuality quality, double clientRate,
double serverRate) {
    if (quality != UNKNOWN && quality != BAD) {
        if (serverRate < 10000) {
            // Video is muted by publisher
        }
    }
}
```