Audio/video track status detection: muted/unmuted

- Receiving stream status after stream subscription

Since IOS SDK build2.6.32it is possible to receive stream publisher events while playing the stream. This can be used to detect if audio or video is muted by stream publisher: when piblisher uses muteAusio()/muteVideo() method, a special event is sending to all the subscribers. To receive this event while playing a stream, define the functionFPWCSApi2Stream.onStreamEvent in Objective C application or WCSStream.onStreamEvent in Swift application.

Objective C examplecode

```
    [_remoteStream onStreamEvent:^(FPWCSApi2StreamEvent *streamEvent){
        NSLog(@"No remote stream, %@", streamEvent.type);
        if ([streamEvent.type isEqual:[FPWCSApi2Model streamEventTypeToString:
kFPWCSStreamEventTypeAudioMuted]]) {
            [_remoteControl onAudioMute:true];
        }
        if ([streamEvent.type isEqual:[FPWCSApi2Model streamEventTypeToString:
kFPWCSStreamEventTypeAudioUnmuted]]) {
            [_remoteControl onAudioMute:false];
        }
        if ([streamEvent.type isEqual:[FPWCSApi2Model streamEventTypeToString:
kFPWCSStreamEventTypeVideoMuted]]) {
            [_remoteControl onVideoMute:true];
        }
        if ([streamEvent.type isEqual:[FPWCSApi2Model streamEventTypeToString:
kFPWCSStreamEventTypeVideoUnmuted]]) {
            [_remoteControl onVideoMute:false];
        }
    }];
```

Swift examplecode

```
            playStream?.onStreamEvent({streamEvent in
                if (streamEvent?.type == FPWCSApi2Model.streamEventType(toString: .
fpwcsStreamEventTypeAudioMuted)) {
                    self.remoteViewController?.onAudioMute(true);
                }
                if (streamEvent?.type == FPWCSApi2Model.streamEventType(toString: .
fpwcsStreamEventTypeAudioUnmuted)) {
                    self.remoteViewController?.onAudioMute(false);
                }
                if (streamEvent?.type == FPWCSApi2Model.streamEventType(toString: .
fpwcsStreamEventTypeVideoMuted)) {
                    self.remoteViewController?.onVideoMute(true);
                }
                if (streamEvent?.type == FPWCSApi2Model.streamEventType(toString: .
fpwcsStreamEventTypeVideoUnmuted)) {
                    self.remoteViewController?.onVideoMute(false);
                }
            });
```

Receiving stream status after stream subscription

When a subscriber connects to a stream to play, this stream status can be received in StreamStatusPlaying event handler using Stream.getAudioState() and Stream.getVideoState() methods

Objective C examplecode

```
    [_remoteStream on:kFPWCSStreamStatusPlaying callback:^(FPWCSApi2Stream *stream){
        [self changeStreamStatus:stream];
        [self onStarted];
        _useLoudSpeaker.control.userInteractionEnabled = YES;
        [_remoteControl onAudioMute:[stream getAudioState].muted];
        [_remoteControl onVideoMute:[stream getVideoState].muted];
    }];
```

Swift examplecode

```
    @IBAction func playPressed(_ sender: Any) {
        changeViewState(playButton,false)
        if (playButton.title(for: .normal) == "PLAY") {
            let options = FPWCSApi2StreamOptions()
            options.name = playName.text;
            options.display = remoteDisplay.videoView;
            options.constraints = remoteMediaConstrains;
            options.transport = tcpTransport.isOn ? kFPWCSTransport.fpwcsTransportTCP : kFPWCSTransport.
fpwcsTransportUDP;
            do {
            playStream = try session!.createStream(options)
            } catch {
                print(error);
            }
            playStream?.on(.fpwcsStreamStatusPlaying, {rStream in
                self.changeStreamStatus(rStream!)
                self.onPlaying(rStream!);
            });
            ...
        }
    }
    ...
    fileprivate func onPlaying(_ stream:FPWCSApi2Stream) {
        playButton.setTitle("STOP", for:.normal)
        changeViewState(loudSpeaker, true)
        changeViewState(playButton, true)
        self.remoteViewController!.onAudioMute(stream.getAudioState()?.muted ?? false)
        self.remoteViewController!.onVideoMute(stream.getVideoState()?.muted ?? false)
    }
```