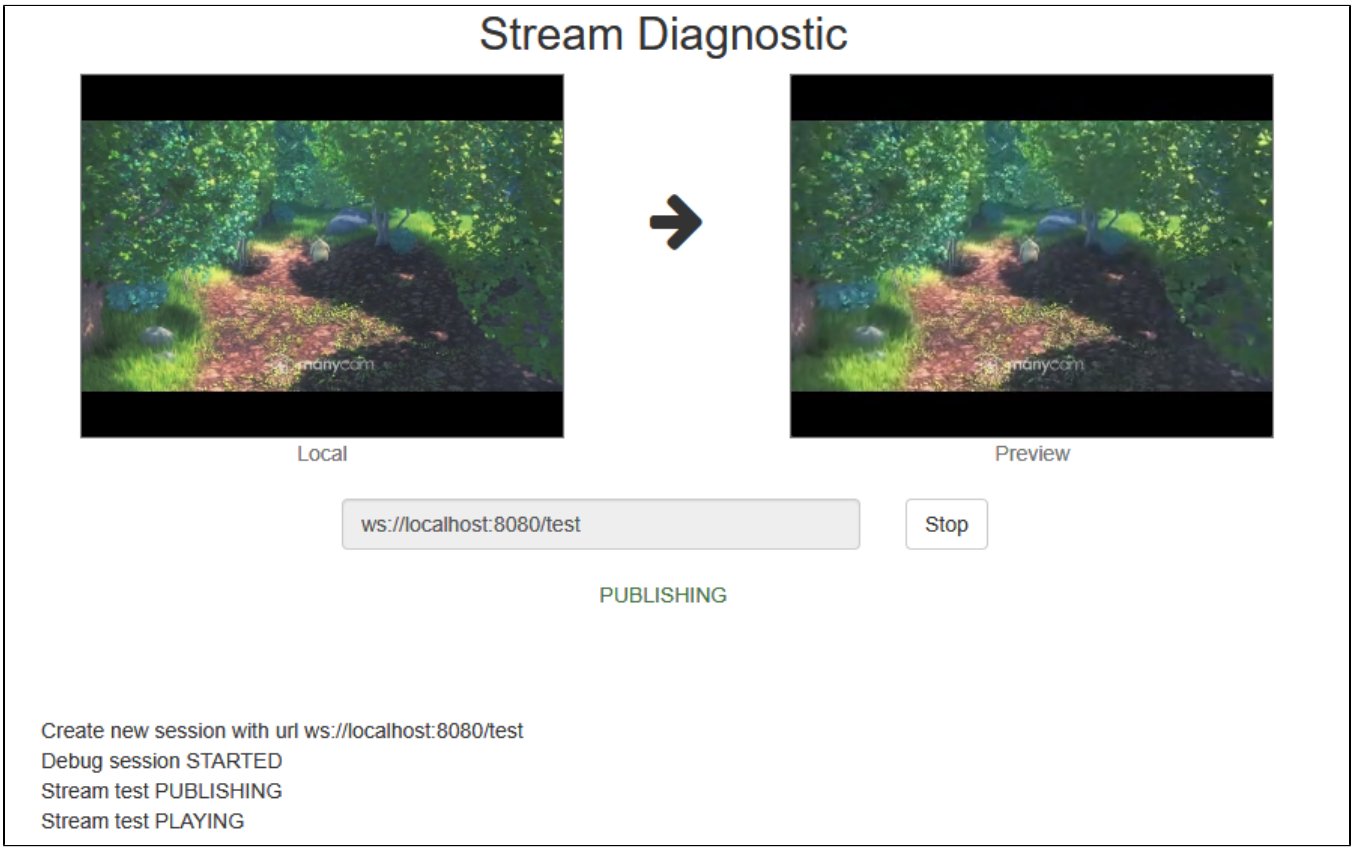
Stream Diagnostic

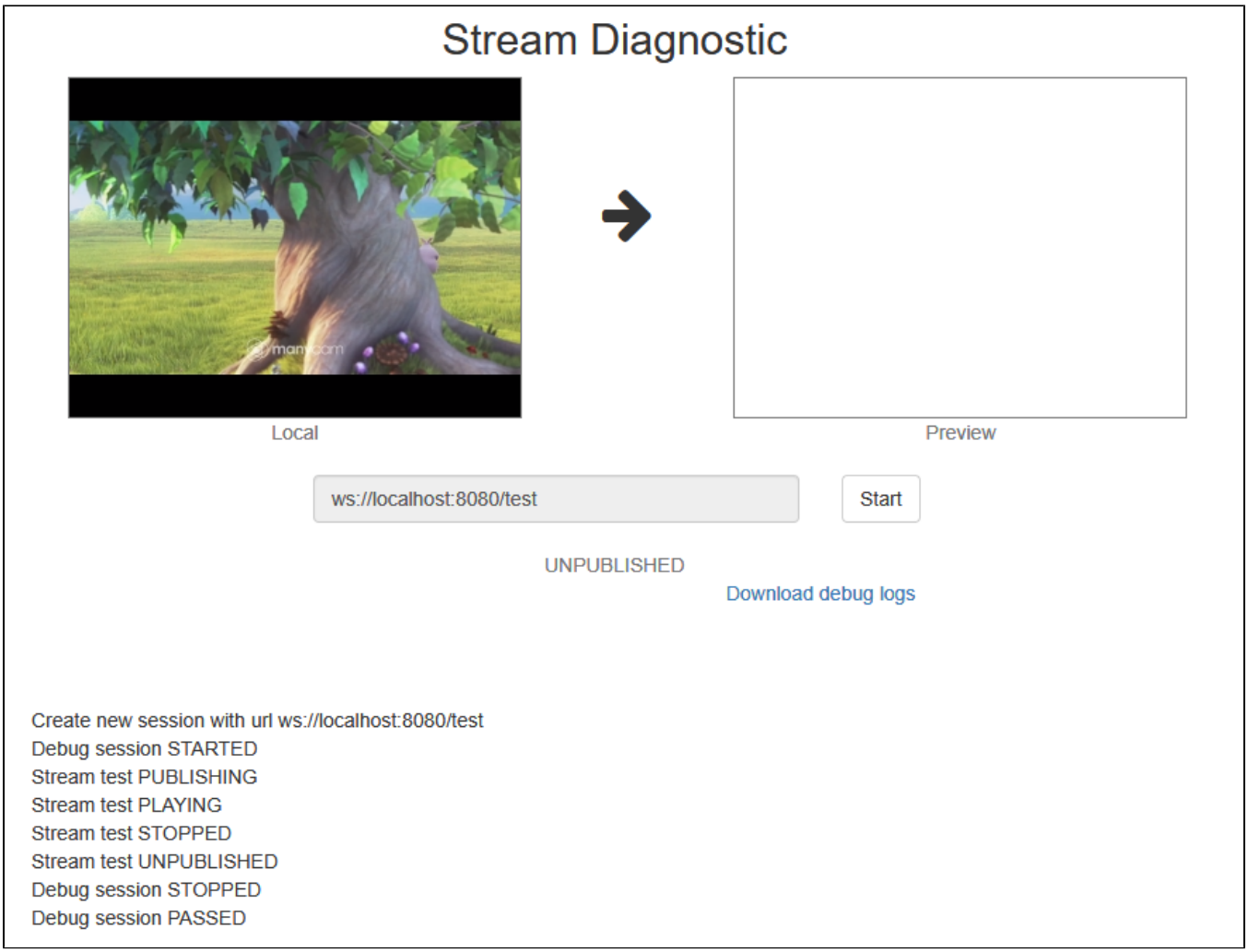
Video streaming and playback with debug information output example

The example shows the possibility to get debug information while streaming and put it on the web page. The sessionDebugEnabled option inwcs-core.properties file **must** be set to true to get debug log and corresponding event in session. This requires server to be restarted.

The example of diagnostic information output while stream is published



and when streaming is stopped



The code of the example

The source code of the example is on WCS server by this path:

/usr/local/FlashphonerWebCallServer/client2/examples/demo/streaming/stream-diagnostic

stream-diagnostic.css - CSS style file
stream-diagnostic.html - the example page
stream-diagnostic.js - script for example to work

The example can be tested at this URL:

https://host:8888/client2/examples/demo/streaming/stream-diagnostic/stream-diagnostic.html

where host is your WCS server address.

Analyzing the code

To analyze the code get stream-diagnostic.js file version with hash ecbadc3 that can be found here and is avalable to download in build 2.0.212.

1. API initializing.

Flashphoner.init() code

```
        Flashphoner.init({createMicGainNode: false});
```

2. Connection to the server

Flashphoner.createSession() code

Every action will be printed to the special page element by function log() call

```
        var url = field('url');
        log("Create new session with url " + url);
        $('#url').prop('disabled', true);
        session = Flashphoner.createSession({urlServer: url}).on(SESSION_STATUS.ESTABLISHED, function(session){
            ...
        });
```

3.Receiving the event confirming successful connection

ConnectionStatusEvent ESTABLISHED code

```
        session = Flashphoner.createSession({urlServer: url}).on(SESSION_STATUS.ESTABLISHED, function(session){
            //session connected, start streaming
            startStreaming(session);
        }).on(SESSION_STATUS.DISCONNECTED, function(){
            ...
        }).on(SESSION_STATUS.FAILED, function(){
            ...
        }).on(SESSION_STATUS.DEBUG, function(event){
            ...
        });
```

4. Start debug output and video streaming

session.startDebug(), session.createStream(), publish() code

On stream creation these parameters are passed:

- streamName - the stream name
- localVideo - -element to display video from web camera

```
    session.startDebug();
    session.createStream({
        name: streamName,
        display: localVideo,
        cacheLocalResources: true,
        receiveVideo: false,
        receiveAudio: false
        ...
    }).publish();
```

5.Receiving the event confirming successful streaming

StreamStatusEvent PUBLISHING code

On this event, preview stream is created with createStream(), and play() called to play it

```
    session.createStream({
        ...
    }).on(STREAM_STATUS.PUBLISHING, function(publishStream){
        log("Stream " + streamName + " " + STREAM_STATUS.PUBLISHING);
        setStatus(STREAM_STATUS.PUBLISHING);
        //play preview
        session.createStream({
            name: streamName,
            display: remoteVideo
            ...
        }).play();
    }).on(STREAM_STATUS.UNPUBLISHED, function(){
        ...
    }).on(STREAM_STATUS.FAILED, function(stream){
        ...
    }).publish();
```

6. PreviewsStream playback stop

previewStream.stop() code

```
function onStarted(publishStream, previewStream) {
    $("#publishBtn").text("Stop").off('click').click(function(){
        $(this).prop('disabled', true);
        previewStream.stop();
    }).prop('disabled', false);
    $("#downloadDiv").hide();
}
```

7.Receiving the event confirming successful playback stop

StreamStatusEvent STOPPED code

```
        session.createStream({
            name: streamName,
            display: remoteVideo
        }).on(STREAM_STATUS.PLAYING, function(previewStream){
            ...
        }).on(STREAM_STATUS.STOPPED, function(){
            log("Stream " + streamName + " " + STREAM_STATUS.STOPPED);
            publishStream.stop();
        }).on(STREAM_STATUS.FAILED, function(stream){
            ...
        }).play();
```

8. Streaming stop when playback is stopped

publishStream.stop() code

```
        session.createStream({
            name: streamName,
            display: remoteVideo
        }).on(STREAM_STATUS.PLAYING, function(previewStream){
            ...
        }).on(STREAM_STATUS.STOPPED, function(){
            log("Stream " + streamName + " " + STREAM_STATUS.STOPPED);
            publishStream.stop();
        }).on(STREAM_STATUS.FAILED, function(stream){
            ...
        }).play();
```

9.Receiving the event confirming successful streaming stop

StreamStatusEvent UNPUBLISHED code

```
    session.createStream({
        name: streamName,
        display: localVideo,
        cacheLocalResources: true,
        receiveVideo: false,
        receiveAudio: false
    }).on(STREAM_STATUS.PUBLISHING, function(publishStream){
        ...
    }).on(STREAM_STATUS.UNPUBLISHED, function(){
        setStatus(STREAM_STATUS.UNPUBLISHED);
        log("Stream " + streamName + " " + STREAM_STATUS.UNPUBLISHED);
        //enable start button
        onStopped();
    }).on(STREAM_STATUS.FAILED, function(stream){
        ...
    }).publish();
```

10. Debug information output stop when streaming is stopped

session.stopDebug() code

```
function onStopped() {
    ...
    if (session)
        session.stopDebug();
}
```

11. Receiving the event the end of debugging session

ConnectionStatusEvent DEBUG code

On this event, the link to download session log is forming

```
        session = Flashphoner.createSession({urlServer: url}).on(SESSION_STATUS.ESTABLISHED, function(session){
            ...
        }).on(SESSION_STATUS.DISCONNECTED, function(){
            ...
        }).on(SESSION_STATUS.FAILED, function(){
            ...
        }).on(SESSION_STATUS.DEBUG, function(event){
            log("Debug session " + event.status);
            if (event.file) {
                var link = window.location.protocol + "//" + window.location.host + "/" + event.file;
                $("#link").attr("href", link);
                $("#downloadDiv").show();
            }
        });
```

12. Debug information output to the page element

code

```
function log(string) {
    document.getElementById("debug").innerHTML += string + '</br>';
}
```