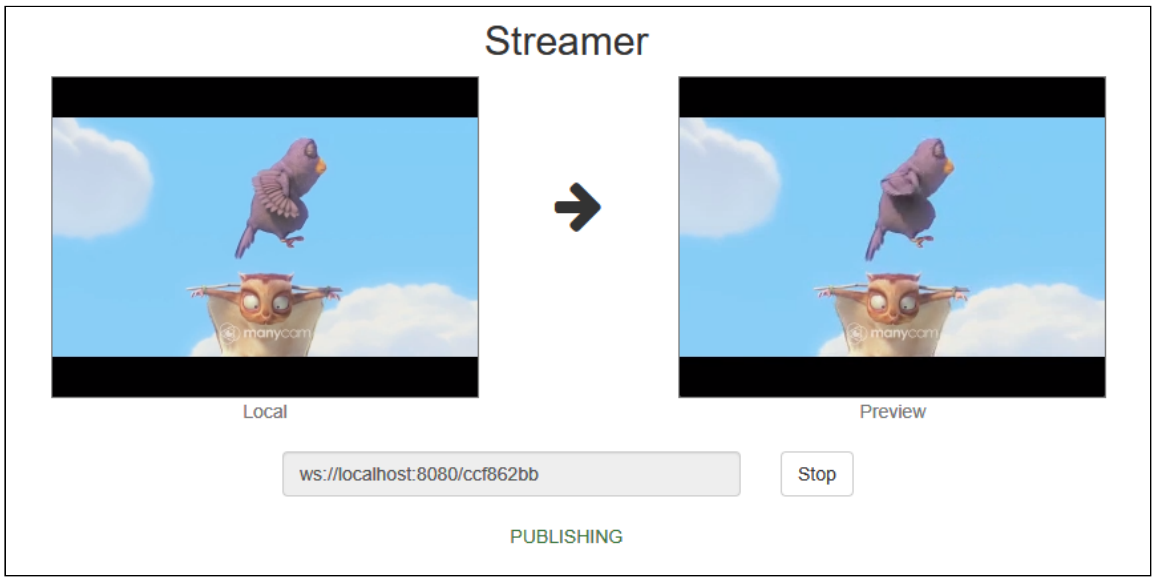
Streamer

Streamer example

This streamer can be used to publish WebRTC stream

On the screenshot below a stream is being published from the client.



In the URL specified in the input field on the screenshot

- `localhost:8080` is the address and Websocket port of the WCS server
- `ccf862bb` is the stream name

Two videos are played on the page

- `Local` - video from the camera
- `Preview` - the video as received from the server

Code of the example

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

/usr/local/FlashphonerWebCallServer/client/examples/demo/streaming/streamer

- steamer.css - file with styles
- steamer.html - page of the streamer

- steamer.js - script providing functionality for the streamer

This example can be tested using the following address:

https://host:8888/client/examples/demo/streaming/streamer/streamer.html

Here host is the address of the WCS server.

Work with code of the streamer

To analyze the code, let's take the version of file `streamer.js` with hash `ecbadc3`, which is available here and can be downloaded with corresponding build 2.0.212.

1. Initialization of the API

`Flashphoner.init()` code

```
Flashphoner.init();
```

2. Connection to server

`Flashphoner.createSession()` code

```
Flashphoner.createSession({urlServer: url}).on(SESSION_STATUS.ESTABLISHED,
function(session){
    ...
}).on(SESSION_STATUS.DISCONNECTED, function(){
    ...
}).on(SESSION_STATUS.FAILED, function(){
    ...
});
```

3. Receiving the event confirming successful connection.

`ConnectionStatusEvent ESTABLISHED` code

```
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(){
    ...
});
```

4. Video streaming

`Session.createStream()`, `Stream.publish()` code

Parameters passed to `createStream()`:

- `streamName` - name of stream;
- `localVideo` - `div` element to display video from camera.

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

5. Receiving the event confirming successful streaming

`StreamStatusEvent PUBLISHING` code

On receiving the event, preview stream is created with `Session.createStream()`, and `Stream.play()` is called to play it.

```
session.createStream({
    name: streamName,
    display: localVideo,
    cacheLocalResources: true,
    ...
}).on(STREAM_STATUS.PUBLISHING, function(publishStream){
    setStatus(STREAM_STATUS.PUBLISHING);
    //play preview
    session.createStream({
        name: streamName,
        display: remoteVideo
        ...
    }).play();
    ...
}).publish();
```

6. Stop of preview playback

`Stream.stop()` code

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

7. Receiving the event confirming playback stop

`StreamStatusEvent STOPPED` code

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

8. Stop of streaming after stop of preview playback

`Stream.stop()` code

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

9. Receiving the event confirming successful streaming stop

`StreamStatusEvent UNPUBLISHED` code

```
session.createStream({
    name: streamName,
    display: localVideo,
    cacheLocalResources: true,
    ...
}).on(STREAM_STATUS.PUBLISHING, function(publishStream){
    ...
}).on(STREAM_STATUS.UNPUBLISHED, function(){
    setStatus(STREAM_STATUS.UNPUBLISHED);
    //enable start button
    onStopped();
}).on(STREAM_STATUS.FAILED, function(){
    ...
}).publish();
```