

Player

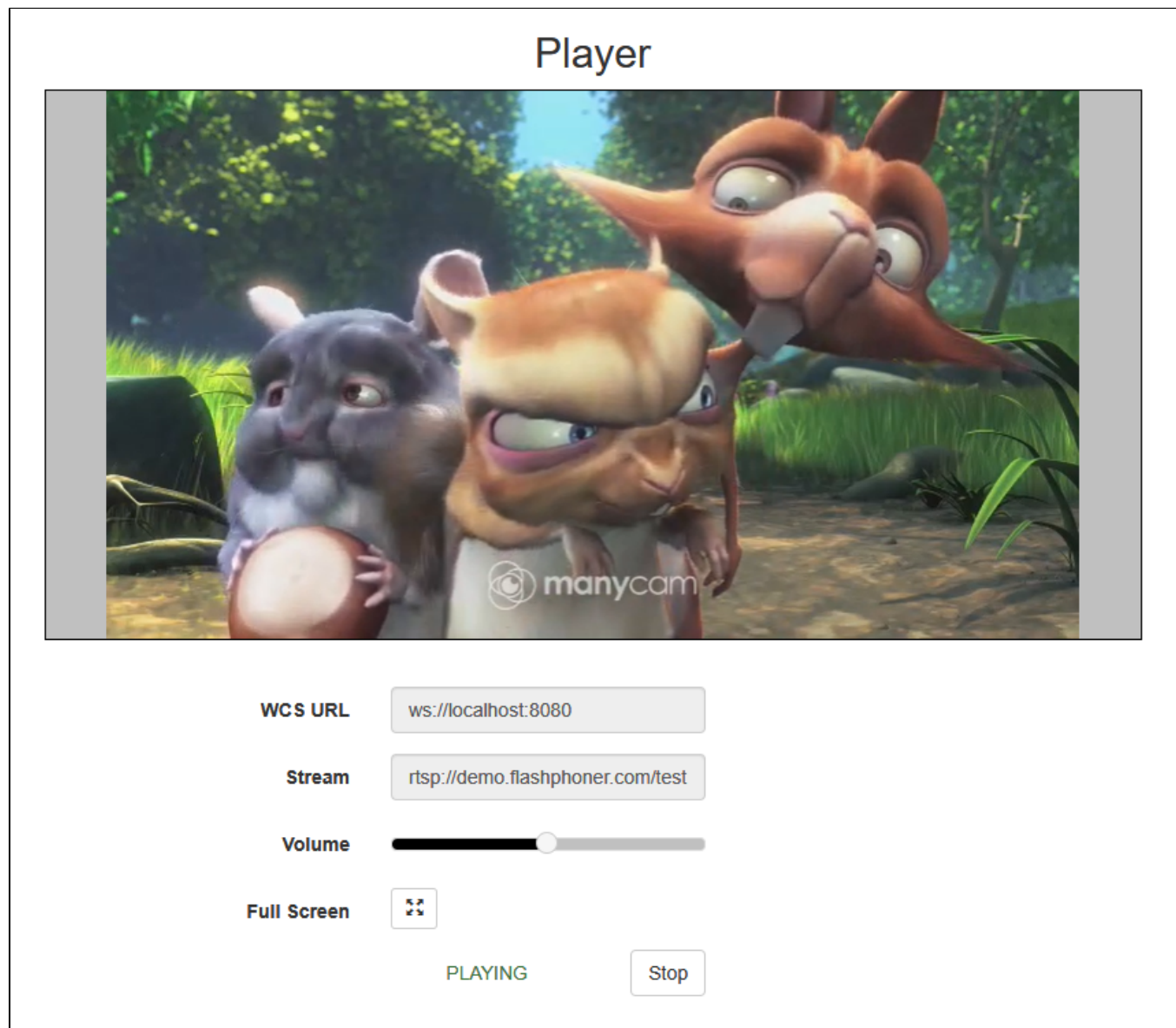
- [Пример плеера](#)
- [Код примера](#)
- [Работа с кодом примера](#)

Пример плеера

Данный плеер может использоваться для воспроизведения любого типа потока с Web Call Server:

- RTSP
- WebRTC
- RTMP

Ниже представлен пример воспроизведения RTSP-потока в web-плеере.



Код примера

Код данного примера находится на сервере по следующему пути:

/usr/local/FlashphonerWebCallServer/client2/examples/demo/streaming/player

player.css - файл стилей
player.html - страница с плеером
player.js - скрипт, обеспечивающий работу плеера

Тестировать данный пример можно по следующему адресу:

<https://host:8888/client2/examples/demo/streaming/player/player.html>

Здесь host - адрес вашего WCS-сервера.

Работа с кодом примера

Для разбора кода возьмем версию файла player.js с хешем 7fff01f, которая находится [здесь](#) и доступна для скачивания в соответствующей сборке 2.0.219.

1. Инициализация API

Flashphoner.init() [code](#)

```
Flashphoner.init({
  receiverLocation: '../dependencies/websocket-player/wsReceiver2.js',
  decoderLocation: '../dependencies/websocket-player/video-worker2.js',
  preferredMediaProvider: mediaProvider
});
```

2. Подключение к серверу

Flashphoner.createSession() [code](#)

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

3. Получение от сервера события, подтверждающего успешное соединение

SESSION_STATUS.ESTABLISHED [code](#)

```
Flashphoner.createSession({urlServer: url}).on(SESSION_STATUS.ESTABLISHED, function(session){
  setStatus(session.status());
  //session connected, start playback
  playStream(session);
}).on(SESSION_STATUS.DISCONNECTED, function(){
  ...
}).on(SESSION_STATUS.FAILED, function(){
  ...
});
```

4. Воспроизведение видеопотока

Session.createStream(), Stream.play() [code](#)

```
stream = session.createStream(options).on(STREAM_STATUS.PENDING, function (stream) {
  ...
});
stream.play();
```

При создании передаются:

- streamName - имя видеопотока
- remoteVideo - div элемент, в котором будет отображаться видео
- разрешение, к которому должна быть транскодирована картинка
- unmutePlayOnStart: false - отключает автоматическое воспроизведение звука при autoplay для соблюдения требований браузеров

[code](#)

```
var options = {
    name: streamName,
    display: remoteVideo,
    flashShowFullScreenButton: true
};
if (Flashphoner.getMediaProviders()[0] === "MSE" && mseCutByIFrameOnly) {
    options.mediaConnectionConstraints = {
        cutByIFrameOnly: mseCutByIFrameOnly
    }
}
if (resolution_for_wsplayer) {
    options.playWidth = resolution_for_wsplayer.playWidth;
    options.playHeight = resolution_for_wsplayer.playHeight;
} else if (resolution) {
    options.playWidth = resolution.split("x")[0];
    options.playHeight = resolution.split("x")[1];
}
if (autoplay) {
    options.unmutePlayOnStart = false;
}
```

5. Получение от сервера события, подтверждающего успешное воспроизведение потока

STREAM_STATUS.PLAYING [code](#)

```
stream = session.createStream(options).on(STREAM_STATUS.PENDING, function (stream) {
    ...
}).on(STREAM_STATUS.PLAYING, function (stream) {
    $("#preloader").hide();
    setStatus(stream.status());
    onStarted(stream);
}).on(STREAM_STATUS.STOPPED, function() {
    ...
}).on(STREAM_STATUS.FAILED, function() {
    ...
});
stream.play();
```

6. Обработка события о недостаточной пропускной способности канала

STREAM_EVENT, STREAM_EVENT_TYPE.NOT_ENOUGH_BANDWIDTH [code](#)

```
}).on(STREAM_EVENT, function(streamEvent){
    if (STREAM_EVENT_TYPE.NOT_ENOUGH_BANDWIDTH === streamEvent.type) {
        var info = streamEvent.payload.info.split("/");
        var remoteBitrate = info[0];
        var networkBandwidth = info[1];
        console.log("Not enough bandwidth, consider using lower video resolution or bitrate. Bandwidth " +
            (Math.round(networkBandwidth / 1000)) + " bitrate " + (Math.round(remoteBitrate / 1000)));
    } else if (STREAM_EVENT_TYPE.RESIZE === streamEvent.type) {
        ...
    }
});
```

7. Обработка события об изменении размера картинки потока

STREAM_EVENT, STREAM_EVENT_TYPE.RESIZE [code](#)

```

    }).on(STREAM_EVENT, function(streamEvent){
        ...
    } else if (STREAM_EVENT_TYPE.RESIZE === streamEvent.type) {
        console.log("New video size: " + streamEvent.payload.StreamerVideoWidth + "x" + streamEvent.payload.
streamerVideoHeight);
    }
});

```

8. Остановка воспроизведения видеопотока.

Stream.stop() [code](#)

```

function onStart(stream) {
    $("#playBtn").text("Stop").off('click').click(function(){
        $(this).prop('disabled', true);
        stream.stop();
    }).prop('disabled', false);
    ...
}

```

9. Получение от сервера события, подтверждающего успешную остановку воспроизведения потока

STREAM_STATUS.STOPPED [code](#)

```

stream = session.createStream(options).on(STREAM_STATUS.PENDING, function (stream) {
    ...
}).on(STREAM_STATUS.STOPPED, function () {
    $("#preloader").hide();
    setStatus(STREAM_STATUS.STOPPED);
    onStopped();
}).on(STREAM_STATUS.FAILED, function() {
    ...
}).play();

```

10. Регулировка громкости воспроизведения

Stream.unmuteRemoteAudio(), Stream.setVolume() [code](#)

```

$("#volumeControl").slider({
    range: "min",
    min: 0,
    max: 100,
    value: currentVolumeValue,
    step: 10,
    animate: true,
    slide: function(event, ui) {
        //WCS-2375. fixed autoplay in ios safari
        if (stream.isRemoteAudioMuted()) {
            stream.unmuteRemoteAudio();
        }
        currentVolumeValue = ui.value;
        stream.setVolume(currentVolumeValue);
    }
}).slider("disable");

```

11. Автозапуск воспроизведения при загрузке страницы

[code](#)

```

if (autoplay) {
    // We can start autoplay with muted audio only, so set volume slider to 0 #WCS-2425
    $("#volumeControl").slider('value', 0);
    $("#playBtn").click();
}

```

