Using ffmpeg

Overview

ffmpeg is a powerful cross-platform tool for processing and publishing video- and audiocontent. In terms of publishing RTMP stream on server, ffmpeg allows:

- to configure stream encoding parameters very flexible;
- to send RTMP connection parameters to the server.

Quick manual on testing

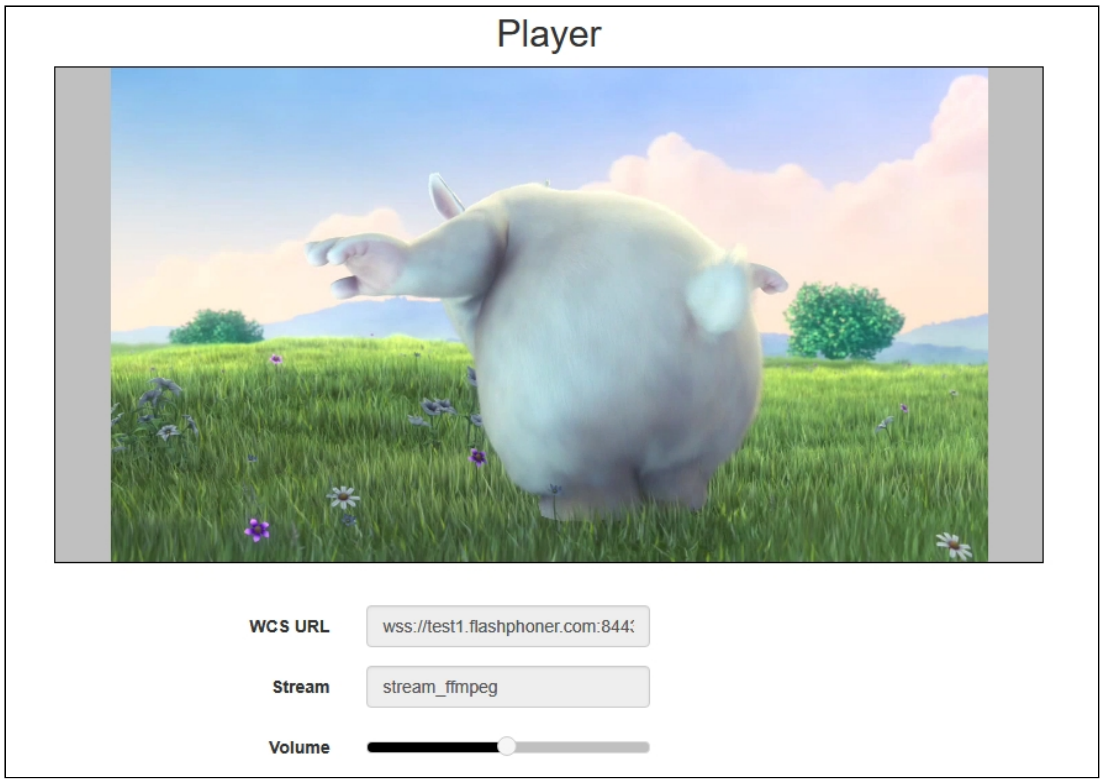
1. For test we use:
2. WCS server
3. ffmpeg
4. Player web application in chrome browser to stream playback
5. Launch ffmpeg

```
ffmpeg -re -i BigBuckBunny.mp4 -preset ultrafast -acodec aac -vcodec h264
-strict -2 -f flv rtmp://test1.flashphoner.com:1935/live/stream_ffmpeg
```

Where

6. `BigBuckBunny.mp4` is a file to publish
7. `test1.flashphoner.com` is WCS server
8. `stream_ffmpeg` is a stream name to publish on server
9. Open Player application in browser `https://test1.flashphoner.com:8888/client2/examples/demo/streaming/player/player.html`, where `test1.flashphoner.com` is WCS server. Set the stream name

and click `Play`. The stream playback begins:



Sorenson Spark + Speex 16 kHz stream publishing

WCS server can capture RTMP stream encoded with Sorenson Spark + Speex 16kHz to FLV container. This stream can be published, for example, using ffmpeg as follows:

```
ffmpeg -re -i BigBuckBunny.flv -preset ultrafast -ar 16000 -ac 1 -acodec
speex -vcodec flv -strict -2 -f flv
rtmp://test1.flashphoner.com:1935/live/test
```

Known limits

1. To handle such stream including stream recording, the stream will be transcoded to H.264 + AAC.
2. Payload types 127 for video and 97 for audio should be set in SDP when publishing such stream, for example

```
v=0
o=- 1988962254 1988962254 IN IP4 0.0.0.0
c=IN IP4 0.0.0.0
t=0 0
a=sdplang:en
m=video 0 RTP/AVP 127
a=rtpmap:127 FLV/90000
a=sendonly
```

```
m=audio 0 RTP/AVP 97 8 0
a=rtpmap:97 SPEEX/16000
a=rtpmap:8 PCMA/8000
a=rtpmap:0 PCMU/8000
a=sendonly
```

The features

Explicit specification of encoding parameters

ffmpeg allows to explicitly set the encoding parameters when stream is publishing, for example

```
ffmpeg -re -i BigBuckBunny.mp4 -acodec aac -vcodec libx264 -f flv -ar 44100
rtmp://127.0.0.1:1935/live/stream_ffmpeg
```

A more complicated case with addition of a sound track to a file that has not one (`/dev/zero` source is used for example):

```
ffmpeg -re -f lavfi -i "movie=filename=test.mov:loop=0,
setpts=N/(FRAME_RATE*TB)" -an -s 1280x720 -r 15 -f rawvideo -pix_fmt yuv420p
- | ffmpeg -thread_queue_size 512 -an -f rawvideo -pix_fmt yuv420p -r 15 -s
1280x720 -i - -vn -f s16le -acodec pcm_s16le -ac 2 -i /dev/zero -r 15 -c:v
libx264 -tune zerolatency -profile:v baseline -g 60 -b:v 960k -s 1280x720 -
c:a aac -b:a 64k -f flv rtmp://127.0.0.1/live/test_video
```

Passing parameters to the server when RTMP connection is establishing

In the `-rtmp_conn` option, ffmpeg allows to set RTMP connection parameters that should be passed to the server when stream is publishing:

```
ffmpeg -re -i BigBuckBunny.mp4 -f flv -rtmp_conn "O:1
NS:appKey:flashStreamingApp NS:name:12121212 NS:stream:12121212 NO:custom:O:1
NS:auth:22222222 NS:stream:3333333 O:0 O:0"
rtmp://test1.flashphoner.com:1935/12121212
```

Here, the following parameters are passed

- server application key `flashStreamingApp`
- server application name `12121212`
- server stream name `12121212`
- custom object with additional data:

```
{
 "auth" : "22222222",
```

```
  "stream" : "3333333"
}
```

How to rotate stream published from ffmpeg

ffmpeg RTMP encoder allows to send orientation metadata to WCS server using command line switches:

```
ffmpeg -i input.mp4 -metadata:s:v rotate=90 -vcodec copy -acodec copy -strict
-2 -f flv rtmp://test1.flashphoner.com:1935/live/stream_ffmpeg
```

Note that ffmpeg sends orientattion value but not angle itself.