LiveVoice

Import from GStreamer

Tips and commands for importing with GStreamer to LiveVoice.

Advanced | Ingest audio | Live Translation | Silent Conferencing | Audio Description | Admin

If you are using the importing feature of LiveVoice you can use for example GStreamer to send audio and video to LiveVoice. There are also tools like FFmpeg, but we recommend GStreamer for its versatility and overall flexibility.

Preparation

1) Set up LiveVoice event

We assume you already have an Event (set on PRO) with at least one channel set up in LiveVoice. In the channel's settings, if you enable "Import Audio" and have "RTP Listen Raw" as Import Type selected, hostname:port combination is shown, such as, e.g., "Send stream to: jvr14.streaming.livevoice.io:17227". In the following examples, the part in front of the colon, "jvr14.streaming.livevoice.io" will be referred to as "" and the part after the colon, "17227" as "".

2) Download GStreamer.

Download: https://gstreamer.freedesktop.org/download/

Windows Users: use the MSVC 64-bit runtime installer. If GStreamer complains about a missing VC++ Runtime, also install the VC++ Runtime package from Microsoft: https://aka.ms/vs/17/release/vc_redist.x64.exe

3) Open terminal and enter your commands.

Open up a Terminal window (cmd on Windows, Terminal.app on OSX) and make sure, that the command gst-launch-1.0 has been either appended to your PATH variable or that you use the full path to the executable. (e.g.C:\gstreamer\1.0\msvc_x86_64\bin\gst-launch-1.0.exe on Windows or /Library/Frameworks/GStreamer.framework/Commands/gst-launch-1.0 on OSX.

For the following examples, we assume that gst-launch-1.0 has been added to the PATH variable.

GStreamer Sample Commands to use with the Import Feature

Play an Audio File (e.g., mp3, ogg, flac,...) via LiveVoice

You can play any audio file type GStreamer "knows" and has a codec for via LiveVoice.

As "" you have to either use just the file name (my_test_audio.mp3) if the file is in your current working directory or the full path (C:\audio\my_test_audio.mp3), if not.

Kopieren

```
gst-launch-1.0 -v filesrc "location=<audio file name>" ! decodebin ! audioconvert ! audioresample ! "audio/x-raw,chan
e=48000" ! queue ! opusenc "bitrate=128000" ! queue ! rtpopuspay ! udpsink "host=<host>" "port=<audio port>"
```

Convert an Audio File (e.g., mp3, ogg, flac,...) to OGG/OPUS and play via LiveVoice

Converting an audio file to the ogg/opus format before playing saves real-time decoding/recoding CPU resources and can help your computer achieve better playback performance, especially in case you are using the LiveVoice import on multiple channels in parallel. Also, in case you're playing back the same file on a loop, you can save CPU resources by first converting it.

a) Convert once:

```
gst-launch-1.0 -v filesrc "location=<audio file name>" ! decodebin ! audioconvert ! audioresample ! "audio/x-raw,chan
e=48000" ! queue ! opusenc "bitrate=128000" ! queue ! oggmux ! filesink ! "location=<audio file name>.ogg"
```

Kopieren

b) Play often:

```
gst-launch-1.0 -v filesrc "location=<audio file name>.ogg" ! oggdemux ! queue ! rtpopuspay ! udpsink "host=<host>" "
io port>"
```

Kopieren

Send your Computer's Default Audio Input (e.g., your microphone) directly to LiveVoice

```
gst-launch-1.0 -v autoaudiosrc ! audioconvert ! audioresample ! "audio/x-raw,channels=2,rate=16000" ! queue ! opus
e=32000" ! queue ! rtpopuspay ! udpsink "host=<host>" "port=<audio port>"
```

Kopieren

You can, of course, also use other audio input devices of your computer. The command gst-device-monitor-1.0 helps you to find the correct audio source GStreamer needs to use. This command, for example, lists all known audio sources on your system:

```
gst-device-monitor-1.0 Audio/Source
```

Kopieren

The output contains a list of devices, each starting with "Device found:". At the end of each section, there is a line starting with "gst-launch-1.0"; e.g. "gst-launch-1.0 pulsesrc device=alsa_input.usb-046d_0825_3368CE00-02.mono-fallback ! ...". You can use this line then to create your own GStreamer command line and use this specific device to stream directly to LiveVoice:

```
gst-launch-1.0 -v pulsesrc device=alsa_input.usb-046d_0825_3368CE00-02.mono-fallback ! audioconvert ! audiores
udio/x-raw,channels=2,rate=16000" ! queue ! opusenc "bitrate=32000" ! queue ! rtpopuspay ! udpsink "host=<host>" "port=<au
dio port>"
```

Kopieren

Play a Video File (e.g. mp4, avi, mkv,...) via LiveVoice

In case your LiveVoice Channel has video and video import with the "RTP Listen Raw" format as import type enabled, you are, similar to the audio import option, shown a hostname:port combination is shown, such as, e.g., "Send stream to: jvr14.streaming.livevoice.io:10527". In the following examples, the part in front of the colon, "jvr14.streaming.livevoice.io" will be referred to as "" and the part after the colon, "10527" as "". N.b.: is usually the same for audio and video; the port differs for audio and video.

To provide reasonable real-time performance while also providing good quality when streaming a video file via LiveVoice, it is strongly recommended to first convert your existing video file to the format the LiveVoice import requires: audio encoded with the Opus codec, video encoded with the VP8 codec.

So, convert the video file to webm format containing Opus-encoded audio and VP8-encoded video streams:

```
gst-launch-1.0 -v webmmux name=mux ! filesink "location=<video file name>.webm" filesrc "location=<video file nam
debin use-buffering=true name=demux demux. ! videoconvert n-threads=4 ! queue ! vp8enc end-usage=vbr target-bitrate=204
8000 token-partitions=3 threads=4 ! queue ! mux.video_0 demux. ! audioconvert ! audioresample ! "audio/x-raw,channels=2,rat
e=48000" ! audiorate ! queue ! opusenc "bitrate=128000" ! queue ! mux.audio_0
```

N.b.: you can adjust the parameter "target-bitrate" to reasonably match the resolution and bitrate of the input video in order to get the best video quality for the smallest possible file size.

Then, send the video (and audio) to LiveVoice:

```
gst-launch-1.0 filesrc location=<video file name>.webm ! matroskademux name=demux demux. ! queue max-size-tin
sebin ! queue ! rtpvp8pay ! udpsink "host=<host>" "port=<video port>" demux. ! queue max-size-time=0 ! opusparse ! queue ! r
tpopuspay ! udpsink "host=<host>" "port=<audio port>" async=false
```

Send your Computer's Default Video Input (e.g. your webcam) directly to LiveVoice

```
gst-launch-1.0 -v autovideosrc ! videoconvert n-threads=4 ! queue ! vp8enc end-usage=vbr deadline=2 cpu-used=5 a
=true token-partitions=3 threads=4 ! queue ! rtpvp8pay ! udpsink "host=<host>" "port=<video port>"
```

You can, of course, also use other video input devices of your computer. The command gst-device-monitor-1.0 helps you find the correct audio source GStreamer needs to use. This command, for example, lists all known audio sources on your system:

```
gst-device-monitor-1.0 Video/Source
```

The output contains a list of devices, each starting with "Device found:". At the end of each section, there is a line starting with "gst-launch-1.0"; e.g., "gst-launch-1.0 v4l2src ! …". You can use this line then to create your own GStreamer command line and use this specific device to stream directly to LiveVoice:

```
gst-launch-1.0 -v v4l2src ! videoconvert n-threads=4 ! queue ! vp8enc end-usage=vbr deadline=2 cpu-used=5 auto-a
token-partitions=3 threads=4 ! queue ! rtpvp8pay ! udpsink "host=<host>" "port=<video port>"
```

You can also add video options depending on what format options your video source supports (also listed in the output of gst-device-monitor-1.0):

```
gst-launch-1.0 -v v4l2src ! "video/x-raw,width=800,height=600,framerate=20/1" ! videoconvert n-threads=4 ! queue ! 
d-usage=vbr deadline=2 cpu-used=5 auto-alt-ref=true token-partitions=3 threads=4 ! queue ! rtpvp8pay ! udpsink "host=<host>
" "port=<video port>"
```

Send your Computer's Default Audio+Video Input (e.g., your microphone+webcam) directly to LiveVoice

```
gst-launch-1.0 -v autovideosrc ! videoconvert ! vp8enc end-usage=vbr deadline=2 cpu-used=5 auto-alt-ref=true toke
s=3 threads=4 ! queue ! rtpvp8pay ! udpsink "host=<host>" "port=<video port>" autoaudiosrc ! audioconvert ! audioresample ! "
audio/x-raw,channels=2,rate=16000" ! queue ! opusenc "bitrate=32000" ! queue ! rtpopuspay ! udpsink "host=<host>" "port=<a
udio port>"
```

Screencast directly to LiveVoice

Windows:

Kopieren

```
gst-launch-1.0 -v dx9screencapsrc ! videoconvert chroma-mode=GST_VIDEO_CHROMA_MODE_NONE dither=GS
_DITHER_NONE matrix-mode=GST_VIDEO_MATRIX_MODE_OUTPUT_ONLY n-threads=4 ! queue ! vp8enc cpu-used=16
max-quantizer=17 deadline=2 threads=4 static-threshold=100 buffer-size=20000 ! queue ! rtpvp8pay ! udpsink "host=<host>" "
port=<video port>"
```

Linux:

Kopieren

```
gst-launch-1.0 -v ximagesrc ! videoconvert chroma-mode=GST_VIDEO_CHROMA_MODE_NONE dither=GST_VID
ER_NONE matrix-mode=GST_VIDEO_MATRIX_MODE_OUTPUT_ONLY n-threads=4 ! queue ! vp8enc cpu-used=16 max-qu
antizer=17 deadline=2 threads=4 static-threshold=100 buffer-size=20000 ! queue ! rtpvp8pay ! udpsink "host=<host>" "port=<vi
deo port>"
```

OSX:

Kopieren

```
gst-launch-1.0 -v avfvideosrc capture-screen=true ! videoconvert chroma-mode=GST_VIDEO_CHROMA_MODE_N
=GST_VIDEO_DITHER_NONE matrix-mode=GST_VIDEO_MATRIX_MODE_OUTPUT_ONLY n-threads=4 ! queue ! vp8enc c
pu-used=16 max-quantizer=17 deadline=2 threads=4 static-threshold=100 buffer-size=20000 ! queue ! rtpvp8pay ! udpsink "ho
st=<host>" "port=<video port>"
```