ASKED IN COMMUNITY (/Network/Community/QuestionsBrowse?tech=1)

Record screen with input and output audio using with C# SharpAvi and NAudio

Question

Record screen with input and output audio using with C# SharpAvi and NAudio

* (https://stackoverflow.com/users/9591188)

691 0 0

I m trying to making an application which can record the screen and at the same time can record the input and output audio. I checked so many things on web but couldt find anything that can do it together.

I tried to do something with sharpavi and naudio. The code at the bottom can record the

I tried to do something with sharpavi and naudio. The code at the bottom can record the screen with the voice that come from microphone and also record the voice that come from speakers too. The problem is screen video and audio that come from microphone creates a video file and the voice that coming from spearkes creates other mp3 file. (if my boss would want to do something this way i couldnt do :)

So i want create my video file with included sreeen record, input and output voice together. I hope you can help me.

```
        private readonly int screenWidth;
        private readonly int screenHeight;

        private readonly AviWriter writer;
        private readonly IAviVideoStream videoStream;
        private readonly IAviAudioStream audioStream;

        private readonly WaveInEvent audioSource;
        private readonly Thread screenThread;

        private readonly ManualResetEvent stopThread = new ManualReset
Event(false);
        private readonly AutoResetEvent videoFrameWritten = new AutoRe
setEvent(false);
        private readonly AutoResetEvent audioBlockWritten = new AutoRe
setEvent(false);

        WasapiLoopbackCapture capture = new WasapiLoopbackCapture();
        WaveFileWriter writerx;

        public Recorder(string fileName,
            FourCC codec, int quality,
            int audioSourceIndex, SupportedWaveFormat audioWaveFormat,
bool encodeAudio, int audioBitRate)
        {
            System.Windows.Media.Matrix toDevice;
            using (var source = new HwndSource(new HwndSourceParameter
s()))
            {
                toDevice = source.CompositionTarget.TransformToDevice;
            }

            screenWidth = (int)Math.Round(SystemParameters.PrimaryScre
enWidth * toDevice.M11);
            screenHeight = (int)Math.Round(SystemParameters.PrimaryScr
eenHeight * toDevice.M22);

            // Create AVI writer and specify FPS
            writer = new AviWriter(fileName)
            {
```

```
                FramesPerSecond = 10,
                EmitIndex1 = true,
            };

            // Create video stream
            videoStream = CreateVideoStream(codec, quality);
            videoStream.Name = "Screencast";


            var outputFolder = Path.Combine(Environment.GetFolderPath
(Environment.SpecialFolder.Desktop), "NAudio");
            Directory.CreateDirectory(outputFolder);
            var outputFilePath = Path.Combine(outputFolder, "recorded
x.wav");

            writerx = new WaveFileWriter(outputFilePath, capture.WaveF
ormat);

            if (audioSourceIndex >= 0)
            {
                while (capture.CaptureState != NAudio.CoreAudioApi.Cap
tureState.Stopped)
                {
                    Thread.Sleep(500);
                }


                var waveFormat = ToWaveFormat(audioWaveFormat);
                audioStream = CreateAudioStream(waveFormat, encodeAudi
o, audioBitRate);
                audioStream.Name = "Voice";

                audioSource = new WaveInEvent
                {
                    DeviceNumber = audioSourceIndex,
                    WaveFormat = waveFormat,
                    BufferMilliseconds = (int)Math.Ceiling(1000 / writ
er.FramesPerSecond),
                    NumberOfBuffers = 3,
                };

                audioSource.DataAvailable += audioSource_DataAvailabl
e;
                capture.DataAvailable += Capture_DataAvailable;   //
            }

            screenThread = new Thread(RecordScreen)
            {
```

```
            {
                Name = typeof(Recorder).Name + ".RecordScreen",
                IsBackground = true
            };

            if (audioSource != null)
            {
                videoFrameWritten.Set();
                audioBlockWritten.Reset();
                audioSource.StartRecording();
                capture.StartRecording();//
            }

            screenThread.Start();
        }

        private void Capture_DataAvailable(object sender, WaveInEventA
rgs e)
        {
                writerx.Write(e.Buffer, 0, e.BytesRecorded);
        }

        private IAviVideoStream CreateVideoStream(FourCC codec, int qu
ality)
        {

            if (codec == KnownFourCCs.Codecs.Uncompressed)
            {
                return writer.AddUncompressedVideoStream(screenWidth,
 screenHeight);
            }
            else if (codec == KnownFourCCs.Codecs.MotionJpeg)
            {
                return writer.AddMotionJpegVideoStream(screenWidth, sc
reenHeight, quality);
            }
            else
            {
                return writer.AddMpeg4VideoStream(screenWidth, screenH
eight, (double)writer.FramesPerSecond,

                    quality: quality,
                    codec: codec,
                   forceSingleThreadedAccess: true);
            }
        }

        private IAviAudioStream CreateAudioStream(WaveFormat waveForma
```

```
t, bool encode, int bitRate)
        {

            if (encode)
            {

                return writer.AddMp3AudioStream(waveFormat.Channels, w
aveFormat.SampleRate, bitRate);
            }
            else
            {
                return writer.AddAudioStream(
                    channelCount: waveFormat.Channels,
                    samplesPerSecond: waveFormat.SampleRate,
                    bitsPerSample: waveFormat.BitsPerSample);
            }
        }

        private static WaveFormat ToWaveFormat(SupportedWaveFormat wav
eFormat)
        {
            switch (waveFormat)
            {
                case SupportedWaveFormat.WAVE_FORMAT_44M16:
                    return new WaveFormat(44100, 16, 1);
                case SupportedWaveFormat.WAVE_FORMAT_44S16:
                    return new WaveFormat(44100, 16, 2);
                default:
                    throw new NotSupportedException("Wave formats othe
r than '16-bit 44.1kHz' are not currently supported.");
            }
        }

        public void Dispose()
        {
            stopThread.Set();
            screenThread.Join();

            writerx.Dispose();//
            writerx = null;//
            capture.Dispose();//

            if (audioSource != null)
            {
                audioSource.StopRecording();
                audioSource.DataAvailable -= audioSource_DataAvailabl
e;
            }
```

```
            writer.Close();
            stopThread.Close();
        }

        private void RecordScreen()
        {
            var stopwatch = new Stopwatch();
            var buffer = new byte[screenWidth * screenHeight * 4];

            Task videoWriteTask = null;

            var isFirstFrame = true;
            var shotsTaken = 0;
            var timeTillNextFrame = TimeSpan.Zero;
            stopwatch.Start();

            while (!stopThread.WaitOne(timeTillNextFrame))
            {
                GetScreenshot(buffer);
                shotsTaken++;

                // Wait for the previous frame is written
                if (!isFirstFrame)
                {

                    videoWriteTask.Wait();

                    videoFrameWritten.Set();
                }

                if (audioStream != null)
                {
                    var signalled = WaitHandle.WaitAny(new WaitHandle
[] { audioBlockWritten, stopThread });
                    if (signalled == 1)
                        break;
                }


                videoWriteTask = videoStream.WriteFrameAsync(true, buf
fer, 0,

                timeTillNextFrame = TimeSpan.FromSeconds(shotsTaken /
 (double)writer.FramesPerSecond - stopwatch.Elapsed.TotalSeconds);
                if (timeTillNextFrame < TimeSpan.Zero)
```

```
                timeTillNextFrame = TimeSpan.Zero;

                isFirstFrame = false;
            }

            stopwatch.Stop();

            // Wait for the last frame is written
            if (!isFirstFrame)
            {

                videoWriteTask.Wait();

            }
        }

        private void GetScreenshot(byte[] buffer)
        {
            using (var bitmap = new Bitmap(screenWidth, screenHeight))
            using (var graphics = Graphics.FromImage(bitmap))
            {
                graphics.CopyFromScreen(0, 0, 0, 0, new System.Drawin
g.Size(screenWidth, screenHeight));
                var bits = bitmap.LockBits(new Rectangle(0, 0, screenW
idth, screenHeight), ImageLockMode.ReadOnly, PixelFormat.Format32bppRg
b);
                Marshal.Copy(bits.Scan0, buffer, 0, buffer.Length);
                bitmap.UnlockBits(bits);
            }
        }

        private void audioSource_DataAvailable(object sender, WaveInEv
entArgs e)
        {
            var signalled = WaitHandle.WaitAny(new WaitHandle[] { vide
oFrameWritten, stopThread });
            if (signalled == 0)
            {
                audioStream.WriteBlock(e.Buffer, 0, e.BytesRecorded);
                audioBlockWritten.Set();
            }
        }
```

Source: https://stackoverflow.com/questions/49631380 (https://stackoverflow.com/questions/49631380)

No Answers yet

Hi, Please Login or join to the community to answer