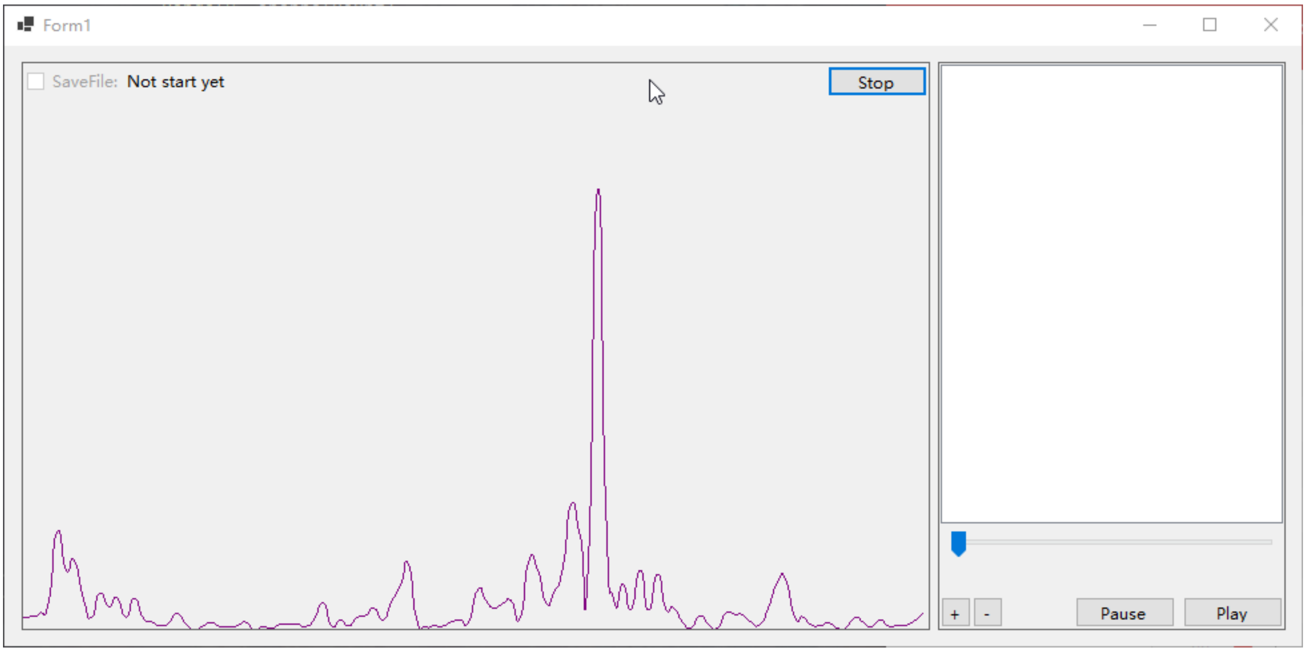
[C #] Using Naudio to achieve audio visualization

tags: .NET (/tag/.NET/)

Preview:



Capture the sound card output:

Achieve audio visualization, the first step is to get audio sampling, here we choose audio that is playing with the computer as a sampling source:

In Naudio, you can use WasapiloPBackcapture to capture:

```
WasapiLoopbackCapture cap = new WasapiLoopbackCapture();
Cap.Dataavailable + = (sender, e) => // Record data When it is available, the audio data is included in the paramete
r.
{
         Float [] allsamples = enumerable // extract the sampling in the data
                 .Range (0, E.BYTESRECORDED / 4) // Disposable by four bytes in the buffer constitute a floating poin
t number, a floating point number is a sample
                 .Select (i => Bitconverter.tosingle (E.BUFFER, I * 4)) // Convert to FLOAT
                 .Toarray (); // Convert to array
         // After obtaining the sample, you will perform detailed processing here.
}
 Cap.StartRecording (); // Start recording
```

Separate left and right channels:

After obtaining the sample, we also need to perform a little processing of the sample, because the captured data is divided by channel, generally left and right channels:

```
// Settings We have saved the number of samples to the variables on allSamples, type float []
 INT CHANELCOUNT = Cap.waveformat.channels; // WasapiloPBackcapture Waveformat specifies the waveform format of the c
urrent sound, which contains the number of channels
float[][] channelSamples = Enumerable
    .Range(0, channelCount)
    .Select(channel => Enumerable
        .Range(0, AllSamples.Length / channelCount)
        .Select(i => AllSamples[channel + i * channelCount])
        .ToArray())
    .ToArray();
```

Take the channel average

After the sample is divided into a sampling of a channel, we can merge it, take the average of the average to facilitate the drawing:

Confidentialité

```
// Settings We have saved the separate sampling to the variable channelsamples, type float [] []
 //, for example, the number of channels is 2, then the sampling of the left channel is channelsamples [0], the right
channel is ChannelSamples [1]
float[] averageSamples = Enumerable
    .Range(0, AllSamples.Length / channelCount)
    .Select(index => Enumerable
        .Range(0, channelCount)
        .Select(channel => ChannelSamples[channel][index])
        .Average())
    .ToArray();
```

Drawing time domain image:

After processing just sampled, you can directly draw it into the window, which is the time domain image, where the simplest line drawing is used.

```
/ / Set the GRAPHICS object for the window, WindowHeight is the height of the display area of the window
 / / Set the average of the channel sampling is averagesamples, type float []
Point[] points = AverageSamples
    .Select((v, i) => new Point(i, windowHeight - v))
         .Toarray (); // convert data into one of the coordinate points
 g.drawlines (Pens.Black, Points); // Connect these points, draw lines
```

Fourier transform:

Naudio also provides a method of fast Fourier transform. With Fourier transform, the time domain data can be converted to frequency domain data, which is what we said.

```
/ / We will make Fourier transform to AverageAgrpes, get a plural array

 // Because of the fast Fourier transform algorithm, the data length is 2 N times,
 Float log = math.ceiling (Math.log (Average, 2)); // Take the number and round up
 Int newlen = (int) Math.Pow (2, log); // Calculate new length
float[] filledSamples = new float[newLen];
 Array.copy (Average, Averagesamples, FilledSamples, Averagesamples.Length); // Copy to New Array
Complex[] complexSrc = filledSamples
         .Select (v => new complex () {x = v}) // converts the sample to a plural
    .ToArray();
 Fastfouriertransform (false, log, complexsrc); // Fourier transform

 // After the transformation, ComplexSRC has been processed, where frequency domain information is stored.
```

Analyze frequency domain information:

For frequency domain information for Fourier transform, it is necessary to use it to be useful, first of all, extract useful information:

Confidentialité

```
In the Fourier transform result of // Naudio, there is no difference in DC components (this makes our handling more c
onvenient), but it also has a conjugate (but the data is sorry, only half is useful)
 // Still use the just complexsrc as the result of the conversion, its type is Complex []

Complex[] halfData = complexSrc
    .Take(complexSrc.Length / 2)
         .ToArray (); /// half of data
float[] dftData = halfData
         .Select (v => math.sqrt (v.x * v.x + v.y * v.y)) // Mode of the number of times
         .Toarray (); // convert the complex result to the frequency amplitude we need

 / / In fact, you can draw this data to the window here, this is already a frequency domain image, but for music visu
alization, some frequencies we don't need
 //, for example, the frequency of 10,000 Hz, we don't have to draw it at all, take the smallest frequency ~ 2500Hz f
oot
 / / For the result of the conversion, the frequency calculation formula between the two data is the sampling rate /
 sample number, then the number we have to take may also be derived from 2500 / (sampling rate / sample number).
int count = 2500 / (cap.WaveFormat.SampleRate / filledSamples.Length);
float[] finalData = dftData.Take(count).ToArray();
```

Draw a frequency domain image:

After getting FinalData after analysis, we can draw them directly, this time uses soft curves to draw

```
/ / Set the GRAPHICS object for the window, Height is the window height
PointF[] points = finalData
    .Select((v, i) => new PointF(i, height - v))
    .ToArray();
 g.drawcurve (Pens.Purple, Points); // Graphics can draw direct drawings
```

Better drawing:

The above time domain and frequency domain image, we all have a brain's index of data as an X coordinate, and the window height minus the data value as Y coordinate, there are two prominent issues:

1. Data may not fill the width of the window or the width range of the window
2. When the data is too large, it will also cause the drawn line to exceed the window height.

The first problem is good to solve, directly allowing the percentage of the length of the data in the index to be equal to the percentage of the X coordinate relative to the width of the window:

\[x = index \div dataLength * windowWidth \]

For the second question, there are two solutions, one is directly weighted weight, such as unified multiplication, making data minus a section, two is a set of functions, such as a log function, after all, LOG functions are higher variables In the case of, the change trend of variable is getting smaller and smaller, we only need to slightly processing this log function, you can directly apply it to data transformation data, so that it does not exceed the window drawing area

In addition, we can also smoot spectrum display (refer to movie transformation), which is probably like this:

1. For example, the result of this Fourier transform is: `{0, 100, 50}` ,
2. The result of the next Fourier transform is: `{100, 0, 0}` ,
3. It can be obtained that the increment is: `{100, -100, -50}` ,

   When updating the result of the transform, we no longer replace the new results directly, but on the basis of the old result, add increment × weight
4. For example, the weight is `0.5` When the actual increment is: `{50, -50, -25}` ,
5. Then the actual new value is: `{50, 50, 25}` ,
6. If the next transformation is still `{100, 0, 0}` , Then we will again `{50, 50, 25}` Approximate to the new value, the weight is still `0.5` The actual increment is: `{25, -25, -12.5}` ,

Note? This increment is half the last increment, it is just a slow-speed, the greater the difference between the new value, the change in the old value, the faster, and they will continue to coincide, so the speed is constantly Slow, form a spectral diagram of deceleration motion.

more content:

More about Naudio, you can see this article:[C #] NAUDIO's various common usage methods Play recording transcoding audio visualization (https://www.cnblogs.com/slimenull/p/14735111.html)

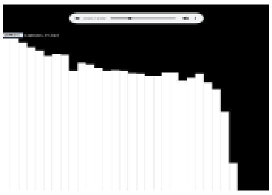
The project has been opened:

About most content involved in this articlegithub.com/SlimeNull/AudioTest (https://github.com/SlimeNull/AudioTest) There is a written in the Null.Audiovisualizer project in the warehouse. (Notes are proper)

In fact, the audio visualization I always want to do it early, but my mathematics is not very good, but finally, I finally persisted, I made it out, my mood is excited.

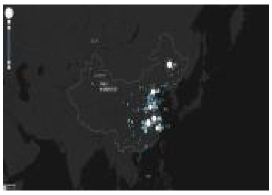
Ask for a look, ask for a comment ~

Intelligent Recommendation


(/article/13012263650/)

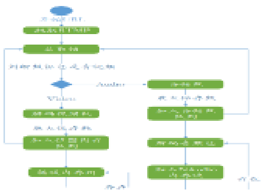
JavaScript audio visualization (/article/13012263650/)

script.js style.css ...


(/article/25141284122/)

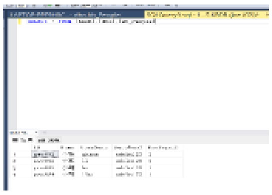
[Tutorial] Gao De Map Using Echarts to achieve data visualization (/article/25141284122/)

About Baidu Map Combined with Echarts to achieve data visualization has been much, after all, it is supported by the official, and will not be described here. Today we will talk about the display of t...


(/article/627578766/)

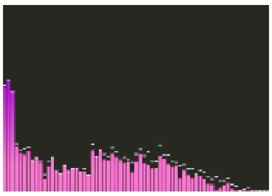
ffmpeg pulls RTMP resources, decodes and plays video and audio Naudio plays audio (/article/627578766/)

Download the development kit. Extraction code: 9yz1 ...


(/article/97262232993/)

C # Using MERGE to achieve synchronization (/article/97262232993/)

SQL MERGE principle and basic usage can be referencedSQLServer Merge Usage If you are synchronized with different library tables of the same server, you can create a job directly according to this art...


(/article/17991041310/)

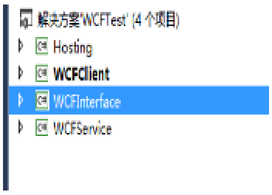
Spectrum visualization: AUDIO API development (/article/17991041310/)

Effect display & Introduction The visual code of this article is mainly based onM I T Open source code from February 15, 2014 MIT spectrum visualization ->Github MIT spectrum visualization diagram...

More Recommendation

C# Use NAudio to get the name of the sound card and play Mp3 (/article/9876773616/)

Namespaces: Get the name of the sound card Play MP3 ...



(/article/289020814/)

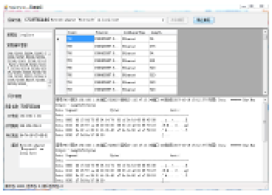
C# practice using WCF to achieve remote calls (/article/289020814/)

Background: Use WCF to implement cross-process ordinary function calls and function calls with callbacks. (For reprint, please indicate the source: cnblogs coder-fang) Solution example diagram:...


(/article/66331998368/)

C # Using Interlocked to achieve thread synchronization (/article/66331998368/)

The process is synchronized by the Interlocked class control counter through the System.ThReading namespace. Some methods for the iterlocked class are as follows: Example, simultaneously open two thre...


(/article/55002328815/)

C # Using Sharppcap to achieve network package capture (/article/55002328815/)

This article uses Sharppcap to realize the captured small example of the network package, implement port monitoring, packet capture, etc., mainly for learning sharing. What is SharppCap? Sharppcap is ...

HTML5-web audio API music visualization (1) (/article/2404654856/)

Use Web Audio API to analyze and manipulate audio, and finally realize an audio visualization program. The final effect pleasePoke here; Full version code pleasePoke here, If you can still see it, ple...

Related Posts

- [C #] Naudio library various common way of use: play recording transcoding audio visualization (/article/47302230254/)
- Naudio audio file conversion (/article/58821599777/)
- C # Based on the Naudio tool to wAV audio file shear (limited to PCM format) (/article/95502312660/)
- C # Naudio Voice (/article/44651157045/)
- Web Audio API for audio visualization (/article/7735799739/)
- C# NAudio record sound and display waveform (/article/6334144052/)
- Play MP3 example with C # and Naudio decoding libraries (/article/34251742152/)
- Python audio visualization (/article/68091301122/)
- Win API audio visualization (/article/8371720118/)
- JavaScript audio visualization (/article/12212076637/)

Confidentialité

Popular Posts

- Java simulates the motion effects of the eight planets (/article/5528776602/)
- Unity globally invoked non-static functions (/article/69082108354/)
- Java Poi exports Excel, the merged cell border disappears (/article/90761684183/)
- P1339 [USACO09OCT] Heat Wave (/article/6980386089/)
- Luo Valley P1783 Beach Defense (/article/56281347267/)
- Realization of sequential traversal methods in binary tree (/article/50341133637/)
- .NET (C #): Judging the inheritance relationship of the TYPE class (/article/51331767555/)
- Beginner MapReduce offline calculation (Eclipse implementation) (/article/63861332534/)
- Send data using Telnet access port (/article/33281997333/)
- PAT (Advanced Level) Practice 1141 PAT Ranking of Institutions (25 points) (/article/1724310546/)

Recommended Posts

- Thread synchronization simple code and Synchronized usage (to be added) (/article/2869829457/)
- Delete a specific type of file under a path, such as a class file, such as a TXT type file. (/article/30951310256/)
- The default initialization value of the Java array element (/article/69861398235/)

LAB6 card (/article/9616725150/)

TP3 Quick Start (/article/20211144167/)

HDU 3089 (Quick Joseph) (/article/14431039988/)

Vuejs experience exchange-form form verification (/article/6743204152/)

Use cookie temporary save (/article/10141600578/)

[Leetcode] delivery information (/article/61272215597/)

CentOS 7.6 Install Apache Tomcat 8.5 (/article/13632240051/)

Related Tags

technical article (/tag/technical+article/)

.NET (/tag/.NET/)

notes (/tag/notes/)

c# (/tag/c%23/)

NAudio (/tag/NAudio/)

C# (/tag/C%23/)

WAV audio file cut (/tag/WAV+audio+file+cut/)

front end (/tag/front+end/)

audio (/tag/audio/)

プ プ グ グ ミ グ グ グ - Python (/tag/%E3%83%97+%E3%83%97+%E3%82%B0+%E3%82%B0+%E3%83%9F+%E3%82%B0+%E3%82%B0+%E3%82%B0+-+Python/)