Compute
Java ASP.NET PHP Oracle PostgreSQL MySQL

< 2016年4月 >

日	一	二	三	四	五	六
27	28	29	30	31	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
1	2	3	4	5	6	7

昵称: Dufe王彬
园龄: 8年
粉丝: 26
关注: 3
+加关注

搜索

找找看

常用链接

我的随笔
我的评论
我的参与
最新评论
我的标签

最新随笔

1. Lua中调用C函数
2. C++ 用libcurl库进行http通讯网络编程(转)
3. SkipList 跳表
4. ntohs, ntohl, htons,htonl的比较和详解【转】
5. SQLITE3 使用总结(转)
6. 软件测试人员必备Linux命令（初、中、高级）
7. epoll用法【整理】
8. 智能指针--C++
9. 关于std:auto_ptr std:shared_ptr std:unique_ptr

WebRtc VoiceEngine代码解析

WebRtc中VoiceEngine可以完成大部分的VOIP相关人物，包括采集、自动增益、噪声消除、回声抑制、编解码、ＲＴＰ传输。下边我们通过代码来解析Voe中处理流程；

创建VoiceEngine和VoEBase

```
[cpp]
VoiceEngine* _vePtr =
VoiceEngine::Create();                          //创建VoiceEngine
VoEBase* _veBasePtr = VoEBase::GetInterface(_vePtr);
//创建VoeBase  所有Voe相关操作通过这个共有类
_veBasePtr-
>Init()
//创建整个Voe处理线程
```

```
VoiceEngine* _vePtr = VoiceEngine::Create();
    //创建VoiceEngine
VoEBase* _veBasePtr = VoEBase::GetInterface(_vePtr);
//创建VoeBase  所有Voe相关操作通过这个共有类
_veBasePtr-
>Init()                                                    //
创建整个Voe处理线程
```

重点就在_veBasePtr->Init() 它会创建voe线程，线程负责采集、数字信号处理、编码、rtp传输。

```
[cpp]
int VoEBaseImpl::Init(AudioDeviceModule* external_adm,
AudioProcessing* audioproc)
{
   _shared->process_thread();    //创建voe线程
   _shared->process_thread()->Start();
   _shared->audio_device()->Init();



}
```

```
int VoEBaseImpl::Init(AudioDeviceModule*
external_adm,   AudioProcessing* audioproc)
{
   _shared->process_thread();    //创建voe线程
   _shared->process_thread()->Start();
```

10. CentOS6.5 一键安装vpn + 添加账号

随笔分类(398)

ADO.NET(1)
Android(12)
AOP(1)
ASP.NET(6)
C Programming(7)
C#.Net(6)
C++(16)
CodeSmith(1)
DIV CSS(41)
EnterPrise Library(1)
ERP分类(1)
Flex(4)
FreeBSD(5)
html5(1)
ipad
iphone
JAVA(10)
JS(30)
Linux(44)
lua(2)
MySQL(21)
NOSQL(1)
Oracle(2)
P2P(1)
Perl(1)
PHP(63)
PostgreSQL(31)
Project Management(1)
SHELL(7)
SQL(13)
UML(3)
Unity3D(4)
webrtc(4)
WebService(2)
xcode(2)
大型系统架构(9)
翻译分类
管理,营销(5)
加密解密(5)
架构设计(5)
软件测试(7)
软件开发(1)
设计模式
数据结构与算法(1)
水晶报表(1)
搜索引擎(5)
网页技术(2)
项目管理(12)

```
  _shared->audio_device()->Init();
```

}audio_device()->Init()重载了int32_t AudioDeviceWindowsWave::Init()（windowns平台），别的平台是别的函数，基本差不多，在这个Init中，创建了ThreadProcess线程，ThreadProcess线程负责所有的音频流程，从设备获取音频数据包。

```
[cpp]
bool AudioDeviceWindowsWave::ThreadProcess()
{
  while ((nRecordedBytes = RecProc(recTime)) > 0);
}

bool AudioDeviceWindowsWave::ThreadProcess()
{
  while ((nRecordedBytes = RecProc(recTime)) > 0);
}处理过程在RecProc

[cpp]
int32_t AudioDeviceWindowsWave::RecProc(LONGLONG&
consumedTime)
{
    _ptrAudioBuffer->DeliverRecordedData(); <SPAN
style="FONT-FAMILY: Arial, Helvetica, sans-serif">}</SPAN>

    int32_t AudioDeviceWindowsWave::RecProc(LONGLONG&
consumedTime)
{
    _ptrAudioBuffer->DeliverRecordedData(); }

[cpp]
int32_t AudioDeviceBuffer::DeliverRecordedData()
{
    _ptrCbAudioTransport->RecordedDataIsAvailable();
}

int32_t AudioDeviceBuffer::DeliverRecordedData()
{
    _ptrCbAudioTransport->RecordedDataIsAvailable();
}
```

RecordedDataIsAvailable是虚函数，被VoeBase重载

```
[cpp]
int32_t VoEBaseImpl::RecordedDataIsAvailable(
       const void* audioSamples,
       uint32_t nSamples,
       uint8_t nBytesPerSample,
       uint8_t nChannels,
       uint32_t samplesPerSec,
```

随笔档案(406)

2016年3月 (3)
2016年1月 (3)
2015年12月 (4)
2015年11月 (1)
2015年10月 (4)
2015年9月 (1)
2015年8月 (1)
2015年7月 (2)
2015年5月 (1)
2015年4月 (8)
2014年8月 (2)
2013年11月 (1)
2013年10月 (3)
2013年3月 (1)
2013年2月 (1)
2013年1月 (1)
2012年2月 (3)
2012年1月 (1)
2011年12月 (7)
2011年11月 (2)
2011年10月 (3)
2011年9月 (19)
2011年8月 (4)
2011年7月 (1)
2011年6月 (3)
2011年5月 (3)
2011年4月 (2)
2011年3月 (2)
2011年2月 (17)
2011年1月 (11)
2010年12月 (24)
2010年11月 (31)
2010年10月 (4)
2010年9月 (14)
2010年8月 (4)
2010年7月 (3)
2010年6月 (1)
2010年5月 (6)
2010年4月 (23)
2010年3月 (1)
2010年2月 (2)
2010年1月 (3)
2009年12月 (3)
2009年11月 (2)
2009年10月 (2)
2009年8月 (13)
2009年7月 (4)
2009年6月 (32)
2009年5月 (16)
2009年4月 (29)
2009年3月 (13)
2009年2月 (8)

```
        uint32_t totalDelayMS,
        int32_t clockDrift,
        uint32_t currentMicLevel,
        bool keyPressed,
        uint32_t& newMicLevel)
{
    _shared->transmit_mixer()->DemuxAndMix();
   _shared->transmit_mixer()->EncodeAndSend();
}

int32_t VoEBaseImpl::RecordedDataIsAvailable(
        const void* audioSamples,
        uint32_t nSamples,
        uint8_t nBytesPerSample,
        uint8_t nChannels,
        uint32_t samplesPerSec,
        uint32_t totalDelayMS,
        int32_t clockDrift,
        uint32_t currentMicLevel,
        bool keyPressed,
        uint32_t& newMicLevel)
{
    _shared->transmit_mixer()->DemuxAndMix();
   _shared->transmit_mixer()->EncodeAndSend();
}
```

DemuxAndMix() 从字面意思是分路与混合，这个函数，主要负责AudioProcess的所有过程，包括Aec,Aecm,AGC,DTMF，遍历所有channel;

[cpp]

```
TransmitMixer::DemuxAndMix()
{
   Channel* channelPtr = sc.GetFirstChannel(iterator);
   while (channelPtr != NULL)
   {
      if (channelPtr->InputIsOnHold())
      {
         channelPtr->UpdateLocalTimeStamp();
      } else if (channelPtr->Sending())
      {
         // Demultiplex makes a copy of its input.
         channelPtr->Demultiplex(_audioFrame);
         channelPtr-
>PrepareEncodeAndSend(_audioFrame.sample_rate_hz_);
      }
      channelPtr = sc.GetNextChannel(iterator);
   }

}

TransmitMixer::DemuxAndMix()
{
```

2008年11月 (3)
2008年10月 (14)
2008年9月 (6)
2008年8月 (12)
2008年7月 (18)

文章分类(1)

大型系统架构(1)

相册(3)

Photo(3)

JS

《快品牌》博客！金错刀看商业2.0

《快品牌》博客！金错刀看商业2.0

myext

最新评论

1. Re:简单的PHP+SMARTY分页类

下一页这个地方page+1不是-1...

--Curitis

2. Re:简单的PHP+SMARTY分页类

哥 借用你这个总算搞定了那个smarty的分页，这个里面类里面有个bug，当点击下一页的时候出现的！

--Curitis

3. Re:C++ 16进制转10进制

@xyqing525我也是菜鸟，学点东西，就记上，以后用就看下，要不然不做就忘了。。^_^...

--Dufe王彬

4. Re:C++ 16进制转10进制

向技术大牛学习,看到你的随笔档案,08-15年间的随笔记录,真的膜拜!

--xyqing525

5. Re:mysql自动备份脚本

您好，请教个问题:1.根据所设置的路径，如：/root/，默认保存到那个盘符里面的？2.根据您贴出的脚本，根据自己数据库密码等设置后（好像没有设置数据库名的），执行后出现一个文件

```
  Channel* channelPtr = sc.GetFirstChannel(iterator);
  while (channelPtr != NULL)
  {
     if (channelPtr->InputIsOnHold())
     {
        channelPtr->UpdateLocalTimeStamp();
     } else if (channelPtr->Sending())
     {
        // Demultiplex makes a copy of its input.
        channelPtr->Demultiplex(_audioFrame);
        channelPtr-
>PrepareEncodeAndSend(_audioFrame.sample_rate_hz_);
     }
     channelPtr = sc.GetNextChannel(iterator);
  }

}
```

Channel::Demutiplex(),基本上没有什么具体任务，就是把audioFrame里边的数据拷贝到channel自身, webrtc是client解决方案，对于client只认为有一个audio source,但可以有多个channel，每个channel中都有audio process，所以需要把数据copy到每个channel.

只有就是数据处理 PrepareEncodeAndSend()

[cpp]

```
Channel::PrepareEncodeAndSend(int mixingFrequency)
{
  if (_inputFilePlaying)
  {
     MixOrReplaceAudioWithFile(mixingFrequency); //如果使用了
voeFile::PlayFileAsMic()；则从文件读取10ms数据，并覆盖audio
buffer
  }

  if (_mute)
  {
     AudioFrameOperations::Mute(_audioFrame);//当然如果设置
mutex，则memset 0
  }
  if (_inputExternalMedia)
  {
  _inputExternalMediaCallbackPtr->Process(); //所过设置了
ExternalMedia,自己的audio处理过程，就是在这里调用的
  }
 InsertInbandDtmfTone();                  //添加DTMF音频
 _rtpAudioProc->ProcessStream(&_audioFrame); // 真正的GIPS
牛逼代码，audio process过程: Aec Aecm AGC
}


Channel::PrepareEncodeAndSend(int mixingFrequency)
{
```

$LogFile，用记.....
--CreateKang

阅读排行榜

1. Linux shell脚本的字符串截取(14486)
2. PgSql备份pg_dump与还原手记pg_restore（转）(7707)
3. C# 10进制与16进制相互转换(3831)
4. 在sql中取系统时间?日期?年?(2947)
5. php soap实例讲解(2556)

评论排行榜

1. 电脑前的朋友注意坐姿 --详图解说(6)
2. 客观公正地评价MySQL和PostgreSQL优劣(转)(2)
3. 简单的PHP+SMARTY分页类(2)
4. C++ 16进制转10进制(2)
5. Linux+Nginx+Php架设高性能WEB服务器(1)

推荐排行榜

1. Linux shell脚本的字符串截取(3)
2. [转]百万级访问网站前期的技术准备(1)
3. 英语语法口诀二十一首(1)
4. css自动截取文字 兼容IE firefox Opera(1)
5. Zend_Auth与Zend_Acl访问控制链(1)

```
 if (_inputFilePlaying)
  {
     MixOrReplaceAudioWithFile(mixingFrequency); //如果使用了voeFile::PlayFileAsMic()；则从文件读取10ms数据，并覆盖audio buffer
  }

  if (_mute)
  {
     AudioFrameOperations::Mute(_audioFrame);//当然如果设置mutex，则memset 0
  }
 if (_inputExternalMedia)
 {
 _inputExternalMediaCallbackPtr->Process(); //所过设置了ExternalMedia,自己的audio处理过程，就是在这里调用的
 }
 InsertInbandDtmfTone();                    //添加DTMF音频
 _rtpAudioProc->ProcessStream(&_audioFrame); // 真正的GIPS牛逼代码，audio process过程: Aec Aecm AGC
}
int AudioProcessingImpl::ProcessStream(AudioFrame* frame) 就是上述调用的_rtpAudioProc->ProcessStream();
```

以上是DemuxAndMix（）过程，之后就是EncodeAndSend()过程，至此整个voe数据处理流程分析结束；

关于Audio Process则是另外一个大话题；

总结一下几点：

1. VoeBase提供大部分的对外接口

2. Channel：继承了大部分的音频功能；

分类：webrtc

好文要顶 关注我 收藏该文

Dufe王彬
关注 - 3
粉丝 - 26
+加关注

0 0

(请您对文章做出评价)

« 上一篇：WebRTC源码分析：音频模块结构分析
» 下一篇：webrtc--AudioProcessing的使用

posted on 2015-04-28 14:11 Dufe王彬 阅读(196) 评论(0) 编辑 收藏