

转载machiner1于 2014-05-08 14:15:40 发布1885收藏2

分类专栏:net

 net 专栏收录该内容

0 订阅8 篇文章订阅专栏

ONVIF开发经验

ONVIF开发经验总结..... 1

一、利用gsoap2.8.14生成Onvif相关源代码..... 2

1. 生成onvif.h头文件的方法..... 2

1) wsdl2h相关命令参数..... 2

2) 链接网络生成..... 2

3) 下载到本地生成..... 2

4) 备注说明..... 3

2. 生成onvif源代码..... 3

1) 整理相关的文件..... 3

2) 通过以下命令生成相关源文件..... 3

3) Soapcpp2相关命令参数..... 3

4) 备注说明..... 3

3. 利用gsoap2.8.8生成源代码和gsoap2.8.14生成的差异..... 4

1) typemap.dat文件中需要添加相关信息..... 4

2) wsas.h中无SOAP_ENV__Fault函数..... 5

二、新建设工程、调试代码..... 5

三、设备发现main函数说明..... 5

四、遇到的问题 and 注意事项..... 9

五、经验总结..... 16

一、利用gsoap2.8.14生成Onvif相关源代码

1. 生成onvif.h头文件的方法

产生头文件有两种方法:链接网络生成和本地生成。

1) wsdl2h相关命令参数

- c 产生c语言的代码,否则产生C++
- s 不使用STL代码
- t 指定typemap.dat文件
- o 指定生成的头文件名

2) 链接网络生成

! 将wsdl2.exe和typemap.dat文件放入同一个文件夹

! 利用cmd或批处理执行以下dos命令:

wsdl2h.exe -c -s -t typemap.dat -o onvif.h

<http://www.onvif.org/onvif/ver10/network/wsdl/remotediscovery.wsdl>

<http://www.onvif.org/onvif/ver10/device/wsdl/devicemgmt.wsdl><http://www.onvif.org/onvif/ver20/analytics/wsdl/analytics.wsdl><http://www.onvif.org/onvif/ver10/analyticsdevice.wsdl><http://www.onvif.org/onvif/ver10/media/wsdl/media.wsdl><http://www.onvif.org/onvif/ver10/deviceio.wsdl><http://www.onvif.org/onvif/ver10/display.wsdl><http://www.onvif.org/onvif/ver10/event/wsdl/event.wsdl><http://www.onvif.org/onvif/ver20/imaging/wsdl/imaging.wsdl><http://www.onvif.org/onvif/ver10/recording.wsdl><http://www.onvif.org/onvif/ver10/replay.wsdl><http://www.onvif.org/onvif/ver10/search.wsdl><http://www.onvif.org/onvif/ver10/receiver.wsdl><http://www.onvif.org/onvif/ver20/ptz/wsdl/ptz.wsdl>

3) 下载到本地生成

! 从ONVIF官网上把相关的WSDL文档下载到本地,下载地址(详细参见备注说明),此外还需下载样式表,下载地址见WSDL文档中的schemaLocation。

发布首篇原创文章，
原力分+10，点亮新秀勋章

分类专栏

	Android system	3篇
	linux - socket	5篇
	net	8篇
	linux - usb	1篇
	语法 - C/C++	1篇
	android - 开发环境	4篇
	linux - 消息队列	1篇
	linux - driver	12篇
	linux - thread	2篇
	ios - xcode发布	7篇
	net - web - server	2篇
	linux	8篇
	sensor	4篇
	tools	4篇
	vc	1篇
	ambarella	1篇
	linux - bluetooth	10篇
	android - bluetooth	2篇
	MTK	
	iOS - http	1篇
	iOS	2篇
	电子	1篇

l 将wsdl2.exe、typemap.dath、WSDL文档和样式表放入同一文件夹中。

l 将相关WSDL文档中的样式表引入路径(schemaLocation)修改为本地实际地址, 如:event.wsdl中
schemaLocation="http://www.w3.org/2005/08/addressing/ws-addr.xsd修改为schemaLocation = ws-addr.xsd

l 利用cmd或批处理执行以下命令

```
wsdl2h.exe -c -s -t typemap.dat -o onvif.hremotediscovery.wsdldevicemgmt.wsdl analytics.wsdl analyticsdevice.wsdl media.wsdl  
deviceio.wsdl display.wsdl imaging.wsdl ecording.wsdl replay.wsdl search.wsdl receiver.wsdl ptz.wsdl
```

4) 备注说明

l wsdl2.exe位于gsoap_2.8.14\gsoap-2.8\gsoap\bin

l typemap.dat位于gsoap_2.8.14\gsoap-2.8\gsoap

l wsdl下载地址: <http://www.onvif.org/Documents/Specifications.aspx>

l 各uri之间用空格隔开

l typemap.dat不需要修改

l 链接网络生成方法的优点是不用考虑以上文件对其它文件的依赖关系,不用修改引入路径。该方法的缺点跟网速有关,因此中途可能会中断,如果超过5分钟未生成,可重新执行命令,通过代理上网则无法生成,不推荐用此方法。

l 下载本地生成方法的优缺点和链接网络方法生成相反,代理网推荐使用此方法。

2. 生成onvif源代码

1) 整理相关的文件

把刚生成的onvif.h与soapcpp2.exe、import和custom放入同一文件夹, 其中:

soapcpp2.exe位于gsoap_2.8.14\gsoap-2.8\gsoap\bin

import位于gsoap_2.8.14\gsoap-2.8\gsoap

custom位于gsoap_2.8.14\gsoap-2.8\gsoap

2) 通过以下命令生成相关源文件

```
soapcpp2.exe -c onvif.h -i import
```

3) Soapcpp2相关命令参数

-2 采用SOAP1.2,

-x 不产生xml文件(不推荐使用此命令,因为xml文档很有用)

-I 为引入路径

-C 只产生客户端代码(注意:C是大写)

4) 备注说明

l 需要在onvif.h中加入#import"wsse.h,用来做安全验证

l 需要将import目录下的wsa5.h以下部分注释掉,否则编译时会报soap_xxxx_SOAP_ENV__Fault()函数重复定义。

```
int SOAP_ENV__Fault  
  
(_QName faultcode, // SOAP 1.1  
  
char *faultstring, //SOAP 1.1  
  
char *faultactor, //SOAP 1.1  
  
struct SOAP_ENV__Detail *detail, // SOAP 1.1  
  
struct SOAP_ENV__Code *SOAP_ENV__Code, // SOAP1.2  
  
struct SOAP_ENV__Reason *SOAP_ENV__Reason, // SOAP 1.2  
  
char *SOAP_ENV__Node, // SOAP 1.2  
  
char *SOAP_ENV__Role, // SOAP 1.2  
  
struct SOAP_ENV__Detail *SOAP_ENV__Detail, // SOAP 1.2  
  
void);
```

3. 利用gsoap2.8.8生成源代码和gsoap2.8.14生成的差异

利用gsoap2.8.8生成源代码方法跟gsoap2.8.14基本一致,但需注意以下区别:

1) typemap.dat文件中需要添加相关信息

t ds = "http://www.onvif.org/ver10/device/wsdl"

te v = "http://www.onvif.org/ver10/events/wsdl"

tl s = "http://www.onvif.org/ver10/display/wsdl"

tm d = "http://www.onvif.org/ver10/deviceIO/wsdl"

ti mg = "http://www.onvif.org/ver20/imaging/wsdl"

tr t = "http://www.onvif.org/ver10/media/wsdl"

tp tz = "http://www.onvif.org/ver20/ptz/wsdl"

tr v = "http://www.onvif.org/ver10/receiver/wsdl"

tr c = "http://www.onvif.org/ver10/recording/wsdl"

ts e = "http://www.onvif.org/ver10/search/wsdl"

tr p = "http://www.onvif.org/ver10/replay/wsdl"

ta n = "http://www.onvif.org/ver20/analytics/wsdl"

ta d = "http://www.onvif.org/ver10/analyticsdevice/wsdl"

td n = "http://www.onvif.org/ver10/network/wsdl"

tt = "http://www.onvif.org/ver10/schema"

OASISRecommended prefixes

```
wstnt ="http://docs.oasis-open.org/wsn/b-2"

wsntw  ="http://docs.oasis-open.org/wsn/bw-2"

wsrfbf ="http://docs.oasis-open.org/wsrf/bf-2"

wsrfr  ="http://docs.oasis-open.org/wsrf/r-2"

wsrfrw = "http://docs.oasis-open.org/wsrf/rw-2"

wstop  ="http://docs.oasis-open.org/wsn/t-1"


# WS-Discovery 1.0 remapping

wsdd10__HelloType    =| wsdd__HelloType

wsdd10__ByeType       =| wsdd__ByeType

wsdd10__ProbeType     =| wsdd__ProbeType

wsdd10__ProbeMatchesType =| wsdd__ProbeMatchesType

wsdd10__ProbeMatchType  =| wsdd__ProbeMatchType

wsdd10__ResolveType   =| wsdd__ResolveType

wsdd10__ResolveMatchesType =| wsdd__ResolveMatchesType

wsdd10__ResolveMatchType =| wsdd__ResolveMatchType

# SOAP-ENV mapping

SOAP_ENV__Envelope = struct SOAP_ENV__Envelope { struct SOAP_ENV__Header*SOAP_ENV__Header; XML
SOAP_ENV__Body; }; | struct SOAP_ENV__Envelope

SOAP_ENV__Header   =| struct SOAP_ENV__Header

SOAP_ENV__Fault     =| struct SOAP_ENV__Fault

SOAP_ENV__Detail    =| struct SOAP_ENV__Detail

SOAP_ENV__Code      =|struct SOAP_ENV__Code

SOAP_ENV__Subcode   =| struct SOAP_ENV__Subcode

SOAP_ENV__Reason    =| struct SOAP_ENV__Reason
```

2) **wsa5.h中无SOAP_ENV__Fault函数**

由于wsa5.h中没有SOAP_ENV__Fault因此不会产生代码重复，因此不用注释。

二、新建工程,调试代码

1. 新建一个项目

将上面生成的soapH.h, soapStub.h, wsdd.nsmmap, soapC.c, soapClient.c, 还有位于gsoap-2.8\gsoap 的stdsoap2.c, stdsoap2.h和位于\custom中的duration.c放入工程中, 然后编写main函数,调试代码。

2. 文件主要功能说明

```
wsdd.nsmmap  名空间定义, 服务器端与客户端都要包含它,里面有很多,都是

相同的,只需导入一个进入工程就行

stdsoap2.h    Header _le of stdsoap2.cppruntime library

stdsoap2.c    RuntimeC library with XML parser and run-time support routines soapStub.h    soapH.h    //Main header file to be included
by all client and servicesources

soapC.c       //Serializers and deserializers for the specied datastructures

soapClient.c  //Clientstub routines for remote operations

soapStub.h    Amodi_ed and annotated header file produced from the input header file
```

三、设备发现main函数说明

```
#include <iostream>

#include "wsdd.nsmmap"

#include "soapH.h"

using namespace std;

int main()

{

    /**声明变量*****/

    structsoap *soap;           //soap环境变量

    structwsdd__ProbeType req;   //客户端发送的Probe

    struct__wsdd__ProbeMatches resp; //服务端回的Probematches

    structwsdd__ScopesType sScope; //Probe里面的范围

    structSOAP_ENV__Header header; //SOAP的头

    intresult = 0;               //返回值

    int count = 0;               //获得的设信息备个数

    /**获取guid(windows下叫guid,linux下叫uuid),格式为urn:uuid:8-4-4-4-12,由系统随机,产生**/

    staticchar buf[64] = {0}; //用来保存guid号

    GUID guid; //声明guid为GUID结构体变量,包含4个变量,分别是

        unsigned longData1;

        unsigned short Data2;

        unsigned short Data3;

        unsigned char Data4[ 8 ];

    */
```

```

if (S_OK== CoCreateGuid(&guid)) //如果guid生成成功,则将其转为字符串,保存在buf中
{
    _snprintf(buf,sizeof(buf)
,"um:uuid:%08X-%04X-%04x-%02X%02X-%02X%02X%02X%02X%02X"
, guid.Data1
, guid.Data2
, guid.Data3
, guid.Data4[0], guid.Data4[1]
, guid.Data4[2], guid.Data4[3], guid.Data4[4],guid.Data4[5]
, guid.Data4[6], guid.Data4[7]
);
}

soap = soap_new(); //初始化soap

if(soap==NULL)
{
    return -1;
}

soap_set_namespaces(soap,namespaces); //设置命名空间

soap->recv_timeout = 5; //设置接收Probematches时间,超过5秒钟没有数据就退出

soap_default_SOAP_ENV__Header(soap,&header); //将header设置为soap消息的头属性

/*****给头赋值*****/

header.wsa__MessageID =buf;

header.wsa__To="urn:schemas-xmlsoap-org:ws:2005:04:discovery";

header.wsa__Action="http://schemas.xmlsoap.org/ws/2005/04/discovery/Probe";

soap->header = &header;

/*设置所需寻找设备的类型和范围,二者至少设定一个,否则可能收到非ONVIF设备,出现异常*/

soap_default_wsdd__ScopesType(soap,&sScope);

sScope->__item ="onvif://www.onvif.org"; //设置所需设备的sScope

soap_default_wsdd__ProbeType(soap,&req);

req.Scopes = &sScope;

req.Types ="tdn:NetworkVideoTransmitter";

/*设置所需设备的类型,tdn为命名空间前缀,为wsdd.nsmap文件中("tdn","http://www.onvif.org/ver10/network/wsdl")的tdn,如过不是tdn,而是
其它,如ns1这里也要随之改为ns1*/

//通过组播发送Probe探针,发送成功返回0,否则-1

result = soap_send_wsdd__Probe(soap,"soap.udp://239.255.255.250:3702", NULL, &req);

if(result== -1)
{
    cout<<"soap error:"<<soap->error<<soap_faultcode(soap)

<<"---"<<soap_faultstring(soap)<<endl;

}else

{
    do{

        result = soap_recv_wsdd__ProbeMatches(soap,&resp);

//接收ProbeMatches,成功返回0,否则-1

        if (result== -1)

            {

                cout<<"共发现"<<count<<"个设备"<<endl;

cout<<"soap error:"<<soap->error<<soap_faultcode(soap)

<<"---"<<soap_faultstring(soap)<<endl;

                break;

            }else

            {

                count++;

                cout<<"===== "<<endl;

                cout<<"UUID:"<<""<<resp.wsdd__ProbeMatches->ProbeMatch->

wsa__EndpointReference.Address<<endl;

cout<<"Type:"<<""<<resp.wsdd__ProbeMatches->ProbeMatch->Types<<endl;

                cout<<"Scopes:"<<""<< resp.wsdd__ProbeMatches->

ProbeMatch->Scopes->__item<<endl;

                cout<<"DeviceService Address:"<<""<<resp.wsdd__ProbeMatches->

ProbeMatch->XAddr<<endl;

                cout<<"MetadataVersion:"<<""<<resp.wsdd__ProbeMatches->

ProbeMatch->MetadataVersion<<endl;

            }

        }while(1);

```

```

    }

    /*****清除变量*****/

    soap_destroy(soap); // removedeserialized class instances (C++ only)

    soap_end(soap);      //clean up and remove deserialized data

    soap_done(soap);

    return result;
}

```

四、调试过程遇到的问题 and 注意事项

1. 出现如下语法错误:

error C2143:语法错误 : 缺少"{"(在":"的前面)

error C2059:语法错误 : ";"

error C2143:语法错误 : 缺少"{"(在":"的前面)

需要将工程中的.c文件改成.cpp文件即可。

2. 无法解析的外部命令错误soap_check_faultsubcode

在stdsoap2.h中声明的soap_check_faultsubcode(struct soap *soap)函数在soapC.cpp中未实现, 可在soapC.cpp中添加如下实现:

```

SOAP_FMAC3 const char * SOAP_FMAC4 soap_check_faultsubcode(struct soap *soap)
{
    soap_fault(soap);

    if(soap->version == 2)
    {
        if(soap->fault->SOAP_ENV__Code && soap->fault->SOAP_ENV__Code->SOAP_ENV__Subcode && soap->fault->SOAP_ENV__Code->SOAP_ENV__Subcode)

            return soap->fault->SOAP_ENV__Code->SOAP_ENV__Subcode->SOAP_ENV__Value;

        return NULL;
    }

    return soap->fault->faultcode;
}

```

3. 无法解析的外部命令错误soap_check_faultdetail

在stdsoap2.h中声明的soap_check_faultdetail(struct soap *soap)函数在soapC.cpp中未实现, 可在soapC.cpp中添加如下实现:

```

SOAP_FMAC3 const char * SOAP_FMAC4 soap_check_faultdetail(struct soap *soap)
{
    soap_fault(soap);

    if(soap->version == 2 && soap->fault->SOAP_ENV__Detail)
        return soap->fault->SOAP_ENV__Detail->__any;

    if(soap->fault->detail)
        return soap->fault->detail->__any;

    return NULL;
}

```

4. 出现无法解析的外部符号 _soap_in_xsd__duration

无法解析的外部符号 _soap_in_xsd__duration, 该符号在函数 _soap_getelement中被引用soapC.obj : error LNK2019: 无法解析的外部符号 _soap_out_xsd__duration, 该符号在函数 _soap_putelement中被引用

soapC.obj: error LNK2019: 无法解析的外部符号 _soap_default_xsd__duration, 该符号在函数 _soap_default__tse__FindMetadata中被引用

需要将\custom文件夹下面的duration.h和duration.c导入工程中。

5. 在VS中出现fatal error C1128: 节数超过对象文件格式限制:请使用/bigobj 进行编译的错误

这是由于源代码文件太大的原因, 需添加选项/bigobj, 在项目属性-> C/C++ ->命令行的附加选项中添加/bigobj。

6. 如果是调用soap_call_XXXX_Probe()来实现设备发现时不能发现所有onvif设备

该函数实现过程中只有一次接收过程, 所以无法发现所有的设备的问题。如果使用该函数, 还需要对函数的实现做以下更改:

函数的接收部分, 将原来的XXXX:Response该为YYYY:ProbeMatches,

其中XXXX是.nsmap文件中<http://www.onvif.org/ver10/network/wsdl>所对应的命名空间前缀, YYYY与后面YYYY:ProbeMatchesType中的前缀相同, 都是<http://schemas.xmlsoap.org/ws/2005/04/discovery>所对应的命名空间前缀名。

7. 抓包实验问题

利用gsoap生成的wsdd.nsmap如下:

```

#include "soaph.h"

SOAP_NMAC struct Namespace namespaces[] =
{
    {"SOAP-ENV","http://schemas.xmlsoap.org/soap/envelope/","http://www.w3.org/soap-envelope", NULL},
    {"SOAP-ENC","http://schemas.xmlsoap.org/soap/encoding/","http://www.w3.org/soap-encoding", NULL},
    {"xsi","http://www.w3.org/2001/XMLSchema-instance","http://www.w3.org/XMLSchema-instance", NULL},
    {"xsd","http://www.w3.org/2001/XMLSchema","http://www.w3.org/XMLSchema", NULL},
    {"wsa","http://schemas.xmlsoap.org/ws/2004/08/addressing", NULL, NULL},
    {"wsdd","http://schemas.xmlsoap.org/ws/2005/04/discovery", NULL, NULL},
    {"chan","http://schemas.microsoft.com/ws/2005/02/duplex", NULL, NULL},
    {"wsa5","http://www.w3.org/2005/08/addressing","http://schemas.xmlsoap.org/ws/2004/08/addressing", NULL},
    {"xmime","http://tempuri.org/xmime.xsd", NULL, NULL},
    {"xop","http://www.w3.org/2004/08/xop/include", NULL, NULL},
    {"t","http://www.onvif.org/ver10/schema", NULL, NULL},
}

```

```

{"wsrfbf","http://docs.oasis-open.org/wsrf/bf-2", NULL, NULL},

{"wsrfr","http://docs.oasis-open.org/wsrf/r-2", NULL, NULL},

{"tad","http://www.onvif.org/ver10/analyticsdevice/wsd", NULL, NULL},

{"tan","http://www.onvif.org/ver20/analytics/wsd", NULL, NULL},

{"tdn","http://www.onvif.org/ver10/network/wsd", NULL, NULL},

{"tds","http://www.onvif.org/ver10/device/wsd", NULL, NULL},

{"tev","http://www.onvif.org/ver10/events/wsd", NULL, NULL},

{"wsnfb","http://docs.oasis-open.org/wsn/b-2", NULL, NULL},

{"timg","http://www.onvif.org/ver20/imaging/wsd", NULL, NULL},

{"tls","http://www.onvif.org/ver10/display/wsd", NULL, NULL},

{"tmd","http://www.onvif.org/ver10/deviceIO/wsd", NULL, NULL},

{"tpz","http://www.onvif.org/ver20/ptz/wsd", NULL, NULL},

{"trc","http://www.onvif.org/ver10/recording/wsd", NULL, NULL},

{"trp","http://www.onvif.org/ver10/replay/wsd", NULL, NULL},

{"trt","http://www.onvif.org/ver10/media/wsd", NULL, NULL},

{"trv","http://www.onvif.org/ver10/receiver/wsd", NULL, NULL},

{"tse","http://www.onvif.org/ver10/search/wsd", NULL, NULL},

{NULL,NULL, NULL, NULL}

};

```

1)通过编写面函数之后,调试运行结果如下:

其中http://192.168.106.112:80/onvif/device_service为本地模拟的设备

通过抓包工具获得信息如下:

2)保留以下命名空间,删除其它信息:

```

#include "soapH.h"

SOAP_NMAC struct Namespace namespaces[] =

{

    {"SOAP-ENV","http://schemas.xmlsoap.org/soap/envelope/","http://www.w3.org/soap-envelope", NULL},

    {"SOAP-ENC","http://schemas.xmlsoap.org/soap/encoding/","http://www.w3.org/soap-encoding", NULL},

    {"xsi","http://www.w3.org/2001/XMLSchema-instance","http://www.w3.org/XMLSchema-instance", NULL},

    {"xsd","http://www.w3.org/2001/XMLSchema","http://www.w3.org/XMLSchema", NULL},

    {"wsa","http://schemas.xmlsoap.org/ws/2004/08/addressing", NULL, NULL},

    {"wsdd","http://schemas.xmlsoap.org/ws/2005/04/discovery", NULL, NULL},

    {"wsa5","http://www.w3.org/2005/08/addressing","http://schemas.xmlsoap.org/ws/2004/08/addressing", NULL},

    {"xmime","http://tempuri.org/xmime.xsd", NULL, NULL},

    {"xop","http://www.w3.org/2004/08/xop/include", NULL, NULL},

    {"t","http://www.onvif.org/ver10/schema", NULL, NULL},

    {NULL,NULL, NULL, NULL}

};

```

调试运行结果如下:

抓包工具抓到信息如下:

3)测试工具信息如下

抓包信息如下:

4)多次实验显示:

wsdd.namsp文件太大时,将会被拆包,192.168.106.164将始终不会回消息,减小命名空间大小,保证只发送一个UDP包,192.168.106.164将始终能回消息

5)通过抓包发现,做设备管理功能时客户端已能与设备端通信,只是涉及到安全,设备端没有返回信息

6)在程序运行时,可能会出现一下信息

这是由于Type和Scopes都没有赋值,接收了非ONVIF的设备

它的body中没有我们需要的值,所以在输出时会引起中断

五、经验总结

1. 对于利用gsoap工具实现基于ONVIF标准的功能,尽量按照如下顺序:

Ø 了解所需实现的功能原理,参考<<ONVIF_Core_Specification,_version_2.0.pdf>>

0 了解gsoap工具的使用方法和编程方法,还有文件结构,参考<<gSOAP 2.8.14 User Guide>>,位于gsoap_2.8.14\gsoap-2.8\gsoap\doc\soapdoc2.pdf里面,里面内容很多,可根据需要查找相关内容,如The wsdl2h WSDL and Schema Importer(84), Using the soapcpp2 Compiler and Code Generator(89)SOAP Header Processing(178页),SOAP/XML Over UDP(208页)

根据所需实现的功能查看对应的xml文档(生成源代码时产生的),因为生成的xml文档是客户端和服务端通信时所发送的模板结构,通过它可以了解编码时所需要填充的信息

Ø 了解工程中.h和.cpp的功能

Ø 查看别人写的例子,理解原理,要学会从原理上去分析碰到的问题并解决问题

Ø 自己写代码验证, 事实求是, 替自己负责, 替用户负责。

2. 善于利用抓包工具

从原理上分析问题,能大大提高效率,如果利用测试工具跟客户端通信,然后进行抓包,能构很好的分析出客户端需发送的消息,服务端回的消息,非常利于编码。

下面是别人的问题点：

按照上述步骤，我试了一遍，最后编译提示 SOAP_TYPE_xsd_duration未声明的标示符，不知道为什么，希望有知道的大神解释一下

Re: [NetPosaMoon](#) 2013-11-22 16:07发表
 回复h1131521123: 加入gsoap\custom\duration.c
 gsoap\custom\duration.h, 另外缺什么就加什么。gsoap\plugin和gsoap\custom都是些扩展文件。

 machiner1	关注		👍 1 🗨️ ⭐ 2 💬 0 📁	专栏目录
ONVIF开发经验分享 这是本人开发完ONVIF给部门分享经验的PPT，希望对有需求的同行有帮助				12-17
ONVIF开发总结 ONVIF开发进行了简单的总结和分析，文中包括利用gsoap2.8.14生成Onvif相关源代码、调试代码、设备发现main函数说明、调试过程遇到的问题和注意...				05-20
gSOAP C++移植和开发总结 未灵的博客 xsd_duration = #import "custom/duration.h" xsd__duration 而你在工程中没有加入和编译链接gsoap目录下的custom/duration.c文件。因此解决办法有...				9-23
gSoap编译器生成onvif框架代码_gsoap生成onvif_houge101的博客 打开sdsoap2.c添加头文件 #include "wsdd.nsmapi" 至此onvif框架代码生成完成 注意：如果出现如下错误 duration.c(61,28): error C2059: 语法错误：“<par...				9-14
ONVIF简介 认识了解ONVIF			weixin_44651073的博客	👤 723
onvif开发总结 热门推荐 ONVIF开发经验总结..... 1一、 利用gsoap2.8.14生成O...				👤 1万+
webbservice中xsd和wsdl和soap的含义_soap_xsd 好大的月亮的博客-CSDN博 ... SOAP协议定义了SOAP消息的格式.SOAIP协议是基于HTTP协议的.SOAP也是基于XML和XSD.XML是SOAP的数据编码方式。打个比喻:HTTP就是普通C...				9-18
gsoap入门获取gsoap的错误信息_c++ 调用gsoap 出现很多未定义_10... (&soap, stream)\\ stream.str())返回std::stringstd::cout<< stream.str() <<std::endl;// C++ 直接输出错误信息到控制台std::errorssoap_stream_fault(&soap,s...				9-8
ONVIF网络摄像头(IPC)客户端开发—ONVIF介绍 1.前言：网上已经有很多关于ONVIF开发的资料，这里概括介绍一下ONVIF协议以及介绍一下我自己在开发ONVIF网络摄像头的一些流程和经验，做个开...			Biao	👤 1万+
ONVIF学习-设备搜索的实现（结合MFC） 经过上一篇博客的介绍，相信大家已经搭建起Onvif的开发框架。这篇博客将介绍一个Onvif的简单应用----设备搜索。博主用的IDE是Visual Studio 2005。...			ORC-Lee的博客	👤 4851
onvif开发之一-gsoap基础 无风也流的博客 SOAP_ENV_Fault重置定义_屏蔽掉gsoap-2.8/gsoap/import/wsa5.h中SOAP_ENV_Fault的定义即可。 undefined reference to `soap_in_xsd_duratio...				9-23
ONVIF学习【2】——SOAP介绍以及gSOAP使用 onvif			holly的专栏	👤 839
gSOAP C++移植和开发总结 1.cannot open file "custom/duration.h" for reading: 解决办法：把gsoap2.8.15/gsoap/custom目录拷贝到当前工作目录2.stdsoap2.cpp:8888: undefined ...			铁匠Smith先生的专栏	👤 1万+
gSoap实现ONVIF中xsd__anyType到具体结构类型的转换 上一篇文章已经粗略计划要讨论gsoap关于序列化/解析编程。本文则阐述一下关于gsoap生成代码的一些重要特征方法及使用。如题，下面我们来自ONVIF生...			dianfen0670的博客	👤 154
Onvif开发框架 包含生成Onvif开发框架的工具，以及已经生成好的开发框架，可直接引用进行后续开发！详细生成过程，请参考我的博文（http://blog.csdn.net/saloon_y...				05-29
C# ONVIF接入开发 ONVIF（开放式网络视频接口论坛）是一个全球化的开放式行业论坛，其目标是促进开发和利用基于物理IP的安全产品接口的全球开放标准。ONVIF创建...				11-24
gSOAP搭建ONVIF (C++)客户端开发框架 --windows & ubuntu windows vs2017搭建ONVIF (C++)客户端开发框架(gSOAP 2.8112)			asdafsdgdnh的博客	👤 2262
Onvif 无法解析的外部符号 soap_in_xsd_dateTime问题，解决（一） 无法解析的外部符号问题也困扰我很久，最后才发现是因为没有添加文件问题，我的天呀！！问题描述： 1> 正在创建库 G:\onvif\Onvif_Client\onvifClient...			Feng的博客	👤 3105
编译器出现conflicting types for 某某的错误原因总结 直译就是xxx发生了一种冲突！比如今天发现的这个错误，实属低级！本次错误的原因是：函数没有先声明，便写在了主函数后面！应该是先声明，后定...			diaoyanqin5061的博客	👤 3127
XML 中的常见问题 XML 中的常见问题 Microsoft Corporation 2000年6月7日 目录 一般问题 什么是XML？ 什么是MSXML？ Microsoft XML 分析器能够做什么？ MSXML、...			蝴蝶俊.net	👤 2882
Onvif-ubuntu:基于vif库的Ubuntu。 基于 OpenOnvif 修改流 URI 提取 Onvif-ubuntu 基于 OpenOnvif				06-30
c语言编译报错问题 duration.c:61:81: error: expected ')' before "" token SOAP_FMAC3 void SOAP_FMAC4 soap_default_xsd__duration(struct soap *soap, LONG64 *a) "ar...			u010798513的博客	👤 1356
Gsoap 集成多个ONVIF服务(DeviceMgmt service) 说明： 本文使用Gsoap 2.8.9， VS2010， win7 所有代码可以到这里下载 另外由于时间有限，整个VS2010 solution第一次编译可能不通过，大家再编译一...			butesbutno的专栏	👤 4876
onvif的nmr开发资料 最新发布 ONVIF是开放网络视频接口论坛（Open Network Video Interface Forum）的缩写，是由若干家视频监控设备生产商共同组成的全球性开发论坛。ONVIF的...				08-31
“相关推荐”对你有帮助么？				
<div>😞 非常没帮助</div> <div>😐 没帮助</div> <div>😏 一般</div> <div>😊 有帮助</div> <div>😄 非常有帮助</div>				
关于我们 招贤纳士 商务合作 寻求报道 ☎ 400-660-0108 ✉ kefu@csdn.net 🗣 在线客服 工作时间 8:30-22:00				
公安备案号 11010502030143 京ICP备19004656号 京公网安备 [2020] 1030-165号 经营性网站备案信息 北京互联网违法和不良信息举报中心 家长监护 网络110报警服务 中国互联网举报中心 Chrome商店下载 账号管理规范 版权与免责声明 侵权申诉 出版物许可证 营业执照 ©1999-2023北京创新乐知网络技术有限公司				



machiner1

码龄 19年

暂无认证

22

2万+

75万+

23万+



原创

周排名

总排名

访问

等级

2548

26

26

5

128

积分

粉丝

获赞

评论

收藏





私信

关注

发布首篇原创文章，
原创力+10分，点亮新秀勋章

去发布

搜博文文章

Q

热门文章

win7 您需要 TrustedInstaller 提供的权限才能对此文件进行更改 17890

android-如何从Play.google.com上下载APK文件到电脑 14770

GetLastError错误码中文大全 12509

dbus-send基本用法 11757

Default-568h@2x.png pngcrush caught libpng error 11443

最新评论

内核3.x版本之后设备树机制
鑫鑫缺铂金: 大佬，写的真好阿

Chief Ray Angle
ilyper: LOOK ONE LOOK 学习了

Chief Ray Angle
zhych856: 学习了 感谢楼主的分享

内核3.x版本之后设备树机制
i513536373: 大神 好厉害 能不能出一篇博文讲述如何分析arm lux启动顺序 做arm...

iOS设置动画画面时间
玄冰武士: 这个方法刚才已经试过了，没用，楼主能说的再仔细点吗？

您愿意向朋友推荐“博客详情页”吗？



强烈不推荐



不推荐



一般般



推荐



强烈推荐

最新文章

2020-12-12

2020-12-12

MSMB953 ANDROID8.1平台GT9XX移植

2020年 3篇

2019年 1篇

2018年 2篇

2017年 3篇

2015年 29篇

2014年 50篇

2010年 6篇