Opencv.org下载源码

加入opencv.framework到工程中

头文件：

核心头文件：#import <opencv2/opencv.hpp>

对ios的支持：#import <opencv2/imgcodecs/ios.h>

矩阵帮助类：#import <opencv2/highgui.hpp>//高级GUI绘制

#import <opencv2/core/types.hpp>

注：框架中NO的地方报错，改成CNO

using namespace cv;不然在调用方法时都要加cv::

//将iOS图片转成OpenCV图片(Mat矩阵)

UIImageToMat(UIImage , Mat mat\_image\_src) mat\_image\_src.clone()

宽、高int width=mat\_image\_src.cols int height=mat\_image\_src.rows

创建一个矩形区域范围：Rect2i rect = Rect2i(j, I, level, level)

拿到一个矩形区域：Mat roi = mat\_image\_clone (rect)

拿到像素点rgb的值：Scalar scalar = Scalar(mat\_image\_src.at<Vec3b>(i, j)[0], mat\_image\_clone.at<Vec3b>(i, j)[1], mat\_image\_clone.at<Vec3b>(i, j)[2] )

将处理好的矩形区域数据拷贝到图片上去Mat roiCopy = Mat(rect.size(), CV\_8UC3, scalar); roiCopy.copyTo(roi) //其中CV\_8UC3，每个颜色值是8位，U表示usign C表示char 3表示3个通道RGB CV表示框架的命名空间

将Mat转成IOS图片：MatToUIImage(Mat )

.mm因为用到了C++

将ARGB转RGB : cvtColor(mat\_image\_src, mat\_image\_dst, CV\_RGBA2RGB, 3)//3表示3个通道

opencv android

导入.so动态库armeabi到src

在哪里找, opencv官网下载：ndk-> opencv -> opencv-android-sdk -> sdk -> native -> libs -> armeabi

在CMakeLists.txt中配置.so动态库 并链接

build.gradle中配置 并 同步

创建Shell文件

touch hello.sh

#!/bin/bash

#!表示约定标记，他会告诉系统这个脚本需要什么样子的解释器来执行，即是一种脚本体现

修改文件的权限:

权限结构：

1.文件类型->QQ群类型

2.文件所有者->群主

3.所有者同组用户->管理员

4.其他用户->群成员

查看每一个部分的权限：

文件类型0或d(表示目录) 属主权限123位置 属组权限456 其他用户权限789

ls –l hello.sh

修改权限: chmod a+x ./hello.sh

执行脚本：./hello.sh

shell脚本的语法：

#：表示注释 没有多行注释 只有单行注释

定义变量：

1.定义变量的时候，变量名不需要加$符号,加了表示取变量的值

2.等号两边不能有空格

3.变量名首字母必须是字母(a-z, A-Z) 或 下划线

4.变量名的中间不允许有空格

5.变量中不允许使用标点符号

只读变量：

关键字：readonly 写在变量前面

删除变量unset 变量名 只读变量不能unset

变量类型：

本地变量：作用域是整个bash进程可以使用 name=”Dream”

局部变量：只作用当前代码段 local name=“Andy”

环境变量：作用域是当前的shell进程以及子进程 export name=”Dream”

位置变量：name=${1} age=${2} echo "${name} ${age}" ${0}是脚本文件的名称

特殊变量：${?}：返回上一个命令执行状态的返回值 0:表示执行成功 1:表示程序执行的结果 2:表示程序执行的状态返回码(0-256)

${#}参数个数,可以不加{} ${\*}参数列表 ${@}参数列表,与\*的区别:\*会将所有参数组成一个字符串,@是分开的 ${$}获取当前shell进程ID ${!}执行上一个指令的pid

字符串，单引号或双引号都可以

字符串拼接 info="${name}${age}" info="姓名："${name}

获取字符串长度name="andy" echo ${#name}

截取：${变量名:开始位置:截取长度} name="I have a dream" echo ${name:2:4}不存在越界的问题 截取到末尾${name:2}

字符串的删除：

一：${变量名#要删除的字符串 正则表达式} 作用：从字符串开头(左边)开始匹配要删除的字符串

name="I have a dream" echo ${name#\*d}得到ream

二：${变量名##要删除的字符串 正则表达式} 作用：从右开始..

echo ${name##\*a} 得到m

三：${变量名%删除字符串 正则表达式} 从字符串结尾开始匹配，删除匹配的字符串 echo ${name%h\*}得到I

四：$(变量名%%删除字符串 正则表达式)

${name%%a\*d}

总结：#表示查询方向从左到右，##表示查询方向从右到左，%表示查询方向从右到左，%%表示查询方向从左到右

echo –e 开启转义

不换行：echo -e "iPhoneX\c" echo "dd"

时间：echo `date`

exit：退出，后面的语句不执行

exit 0默认执行成功 exit 1一般性未知错误

数组：数组名=(值1 值2 值3 …)

name=(1 2) echo $name

name=() name[0]=1 name[1]=2 echo ${name[1]}

输出所有数据: echo ${name[@]} ${name[\*]}

数组元素个数：echo ${#name[@]}

传递参数： ./文件名称.sh 参数1 参数2 参数3

算数运算：a=1 b=2 c=`expr $a + $b` echo $c：其他运算都差不多，除了：

乘法c=`expr $a \\* $b`需要加转义 判断等于： $a == $b

布尔运算$a -le $b:左边数是否小于等于右边数; != ; 或运算: -o ; 相等:-eq ; -ne不相等为true ; -gt大于 ; -a与运算 ; -lt小于 ; -ge大于等于 ; !非运算 ; //这里或与非是位运算，与逻辑运算区别

逻辑运算if [ $a -ge 200 ] && [ $b -eq 100 ] then echo "deng" else echo "budeng" fi

字符串运算[ "a" = "b" ] 其它：!= ; 检测字符串长度是否为0(即是否定义,为0是true): -z [ -z "a" ] [ -n ”$val” ]不为空是true//必须加双引号,否则未定义也true 判断是否为空[ $a ]

文件测试运算符-d file检测是否是目录 if [ -d '/Users/luoyuan/Documents/demos' ]

是否可读写：-r $file -w $file 可执行-x $file -f $file检测是否是普通文件,既不是目录,也不是设备文件 -s $file文件是否为空 -e $file检测文件是否存在

流程控制：

if [ 条件 ] then 代码 elif 条件 then 代码 else 代码 fi

for 变量名 in item1 item2 item3 … do 代码 done

for name in "1" "2" do echo ${name} done

val="I do" for name in ${val} do echo ${name} done//输出I换行do

files=/Users/luoyuan/Desktop/\* for name in ${files} do echo ${name} done

for (( i = 0; i < 10; i++ )); do echo $i done

while循环：while(条件)do 代码 done

a=1 while(($a<10)) do echo $a a=`expr $a + 1` done

无限循环：while : do echo "1" done 或while true 或for((;;))do done

switch语句：number=1 case $number in 1 ) echo "1";; 2) echo "2";; esac

until语句：条件为假时退出循坏i=1 until(($i>1)) do echo "1" done

退出循环break break n：退出n重循环，最里层n是1

文件包含：./fileA.sh source ./fileA.sh

cat：作用：1.查看文件内容。2.连接文件。3.创建一个文件或多个文件。4。重定向到输出终端。5.重定向到文件

cat [选项][文件] cat fileA.sh查看内容 cat –n fileA.sh显示文件内容和行号

cat –b fileA.sh和”-n”类似功能,但是只标记有内容的行号

cat –e fileA.sh在每行后面加$, 将多行内容转为一行内容时有用

read接收输入的内容：read name echo "name: ${name}"

read -p "请输入" name echo "name: ${name}"

设置输入超时时间: if read -t 2 -p "请输入" name then echo "name: ${name}" fi

隐藏输入的内容: read -s -p "请输入" pwd echo "pwd: ${pwd}"

读取文件内容并一行行输出cat fileB.sh | while read line do echo "内容: ${line}" done

接收输入多行while read name

printf不会换行,一般用于格式打印 echo会换行,用于标准输出

printf "%-10s %-8s %-4s \n" 姓名 性别 体重kg 其中”%-10s”指一个宽度为10个字符 “-”表示左对其 “s”表示字符串

“%-4.2f”表示一个宽度为4，小数点后2位的小数

输出转义字符： printf " \"<%s>\" \n" "Dream" 这里指引号

脚本函数：function 方法名称(){ }

function testFunc(){ echo "调用方法" } testFunc

传参function testFunc(){echo "$0" echo "$1" echo "$2 } testFunc $1 $2

输入重定向(将文件内容输入到控制台)：wc 名称 wc fileA.sh将fileA.sh中的内容读取到控制台,读取出的内容依次：行数，词数(以空格分开)，文本的字节数

cat < fileA.sh

输出重定向echo "hello" > fileA.sh//将”hello”写入fileA.sh, 把fileA.sh原来的内容替换(为什么叫重定向，就是把原本定向到控制台的内容重定向到fileA.sh)

追加>> echo "hello" >> fileA.sh//把hello加到末尾

管道：有时候我们需要一个程序的输出作为另一个程序的输入 //lymark

expr: val=$(expr $a + $b) val=$[ $a + $b ]

bash: 允许在命令中输入浮点数表达式,然后解释并且计算表达式，最后返回结果

控制台输入bc: 相当于打开计算器 退出quit

结合管道和bc： val=$(echo " 1.56 \* 100 " | bc) echo $val

bc+输入重定向：val1=1.314 val2=0.618 val=$(bc << FG a = ( $val1 \* $val2 ) a \* 1 FG) echo $val

STIDIN标准输入 STIDOUT标准输出

将内容写入文件: who >> fileB.sh > 是全部替换

将错误信息写入文件ls -a a.sh 2> fileB.sh

将错误信息和正确信息写入不同文件ls -al a.sh fileA.sh 1> fileB.sh 2> fileA.sh

将所有信息写入相同文件ls -al a.sh fileA.sh &> fileB.sh

临时重定向: echo "Hello" >&2//定义为错误信息 则执行./hello.sh 2> fileB.sh将Hello写入fileB.sh

将所有内容输出到fileB.sh: exac 1>fileB.sh 执行./hello.sh

输入 exec 0<fileB.sh//从fileB.sh读取内容： exec 0<fileB.sh count=1 while read line do echo $line done 0表示标准输入，1表示标准输出，2表示错误

创建自定义输入exec 3>fileB.sh echo "3" >&3//将3写入fileB.sh

exec 4>&1一开始是3，重定向到1 还原：exec 1>&4 //涉及输出描述符时反向查询，一直找到输出文件为止

exec 4>fileB.sh exec 4>&1 echo "3" >&4//控制台输出3

exec 4>fileB.sh echo "3" >&4 exec 4>&1//3写入fileB.sh

输入exec 0<&6

读写exec 3<> hello.sh

exec 3<>fileA.sh read line <&3 echo $line echo "写入" >&3//这里会追加，因为读的时候指针偏移了

创建文件描述符(作用：这个文件可以读写)：文件内容->fileContent.sh

关闭文件描述符(读写都不允许)：exec 3>&-

lsof命令：路径/usr/sbin/lsof

/usr/sbin/lsof -a -p $$ -d 0,1,2 其中-a表示将两个选项的结果进行拼接，-p表示进程id，$$表示环境变量，-d表示文件描述符(0、1、2)

结果COMMAND PID USER FD TYPE DEVICE SIZE/OFF NODE NAME

bash 676 luoyuan 0u CHR 16,0 0t6397 627 /dev/ttys000：COMMAND表示正在运行的命令名称(取出前9个字符)->由于名字太长我们可以截取，PID->进程ID，USER->进程所属的登录名，FD->文件的描述符以及访问类型(0输入，1输出，2异常，r，w，u读写)，TYPE->文件类型(CHR:表示字符型，BLK表示块型，DIR表示目录，REG表示文件)，DIVICE表示设备号，SIZE/OFF如果存在表示文件大小，NODE表示本地文件的节点号，NAME表示文件名称)

exec 3>fileA.sh exec 6>fileB.sh exec 7<error.sh /usr/sbin/lsof -a -p $$ -d 0,1,2,3,6,7

阻止命令的输出：cat fileA.sh > /dev/null打印到系统临时文件

清空某个文件：cat /dev/null > fileA.sh

创建临时文件:mktemp dream.XXXXXX 后六位是随机算法

在临时目录创建临时文件mktemp -t testfile.XXXXXX 注:-不要打成中文

创建临时目录mktemp -d testfile.XXXXXX

date | tee fileA.sh前一个输出作为后一个的输入，同时打印到控制台

追加date | tee -a fileA.sh

脚本操作数据库outfile='test.sql' IFS=',' while read name age do cat >> $outfile << EOF INSERT INTO t\_test(name, age) VALUES ('$name', '$age'); EOF done < ${1} #报unexpected end错误(已解决:cat >>…不能缩进lymark，要顶着最左边) IFS: read通过IFS进行字符串分割,分割之后会自动给参数赋值

-e exit表示执行数据库命令

mysql test\_ly -u root -p -e 'show tables;': 其中test\_ly是数据库名称

mysql test\_ly -u root -p << JAR update users set name=’$1’ where id=1; JAR #注意单引号

配置默认登录

课堂老师的做法：/usr/local/mysql目录->surport-files->my-default.cnf 复制为my.cnf,添加password = hnb314159

查找my.cnf：mysql -help -verbose | grep -B1 -i “my.cnf”

对于mysql5.7.18以上的版本：直接在/etc下创建my.cnf, 内容在myLearningResource的my.cnf文件中,密码可以搜索password找到

mysql –uroot就可以直接登录

who命令： 结果：1是用户名，2是窗口终端，3是登录时间

mesg: 结果 is y：表示开启了发消息的功能，is n表示没有开启

who –T: +表示可以发，-表示不能

发送消息:write luoyuan ttys001

开启发消息：mesg y

音视频

框架：ffmpeg.org

curl 表示通过Http\ftp下载或上传：source=”ffmpeg-3.4” curl http://ffmpeg.org/releases/ffmpeg-3.4.tar.bz2 | tar xj(x表示解压文件选项,j表示是否需要解压bz2压缩包)

Help Options帮助选项：./configure –help ./configure --list-encoders显示可以用的编码器 -hwaccels硬件加速

Standard options标准选项: --logfile=FILE打印日志 --prefix=PREFIX静态库打包路径 --datadir=DIR在制定目录下安装二进制文件

Liscensing options许可证: --enable-gpl允许使用GPL代码，由此生成库或二进制文件

Configuation options配置备选选项：--disable-static不能构建静态库；--enable-shared构建动态库

.o文件：保存.h/.m/.c文件编译后的结果 `pwd`当前目录

--enable-cross-compile交叉编译 --disable-debug禁止调试模式 --disable-programs禁用程序,不允许建立命令行程序 --disable-doc不需要编译文档 --enable-pic允许编译与位置无关的代码

平台架构：arm64 armv7真机 x86\_64 i386模拟器

库编译系统版本:target=”7.0” compile=”y”允许编译,n表示不允许

`which` yasm判断是否存在yasm这个汇编器 `which` brew

ruby –e “$(curl –fsSL http://…)” \|| exit 1 //安装失败则退出

--arch $arch指定当前编译的架构类型

执行时报错: GNU assembler not found, install/update gas-preprocessor, 解决:将gas-preprocessor.pl放入/usr/local/bin，并赋予可执行权限

总结：最终需要的：ffmpeg/configure target-os(mac系统) cc(编译平台c++) as(汇编程序) 配置(交叉编译,禁用程序(不要创建命令行程序),禁用调试,不需要编译文档,允许建立与位置无关的代码 extra-ldflags平台版本配置extra-cflags prefix静态库输出目录

--enable-avcodec:编解码库,最重要

make –j3 install $EXPORT 多核支持，这里就死记吧

编译好的.a和.h拷贝到工程中

依赖：coreMedia videotoolbox coregrahic audio libiconv libz libbz2

Header Search path， 去library search paths中找，复制过来加上末尾lib改成include,即头文件路径 配置头文件Header Search Path

头文件:<…/avcodec.h>

报错Undefined symbols for architecture arm64: "\_OBJC\_CLASS\_$： 在Build Phases的Compile Sources中加入FFmpegTest.m，即\_OBJC\_CLASS\_$后面接报错的文件

视频解码关键参数：AVFormatContext AVCodecContext AVCodec AVPacket AVFrame SwsContext out\_buffer FILE

音频解码关键参数：AVFormatContext AVCodecContext AVCodec AVPacket AVFrame SwrContext out\_buffer FILE

要安装NDK环境 配置基本环境：出现appcompat-v7:27类似的报错，到Toos->Android->SDK Manager查看支持，将26改成支持的或下载支持

安卓：1.要勾选include C++，并尽量勾选在java中处理c和c++异常，build.gradle->minsdkversion 14

2.导入FFmpeg开发包，就是导入编译后的.so

3.在CMakeList中配置 4.配置app/build.gradle的平台架构类型

native标记特殊方法，不是普通的java方法，而是与NDK进行交互的方法(c/c++)

cpp/native-lib.cpp链接bundle(在CMakeList add\_library)

权限文件(相当于plist)：…manifest

导入之后先rebuild再run

安卓脚本：指定NDK路径，指定编译版本

安卓3.0有问题，用2.3.3

各种音视频文件或其编码格式，根据格式百度吧

整个文件夹的导入,工程模式下才能copy log的窗口是哪个:Android Monitor

Error:(10) undefined reference to 'avcodec\_configuration' CMakeList中别忘了link链接动态库

GLYUV打开.yuv后缀文件 //花屏:不要勾选auto Detect, 自己设置解码用的屏幕宽高，选择4:2:0，Custom

报错：cannot find –lpostproc。原因:CMakeList跟着课堂抄,结果没有postproc

pcm：音频采样数据格式, aac压缩数据格式

avformat\_open\_input返回-13。manifest没有开权限

x264.nl下载编码器

视频编码的关键参数packet avcodeContext avFormatContext->pb avFormatContext 其他av\_frame out\_buffer

h264播放器: Wondershare Player

ERROR: libx264 not found ：--extra-cflags后面接的x264路径最好用英文

报错: Out of tree… 删除x264中的config.h/x264\_config.h

adb pull 安卓目录 电脑目录

音频解码注意out\_sample\_fmt = AV\_SAMPLE\_FMT\_S16

音频编码的fdk\_aac包下载地址:

http://www.linuxfromscratch.org/blfs/view/svn/multimedia/fdk-aac.html

音频编码关键参数: AVFormatContext AVCodecContext AVCodec out\_buffer FILE AVFrame AVPacket

各个参数的理解:

视频解码:

AVFormatContext:关联封装文件、AVCodecContext、AVPacket的桥梁

AVCodecContext:关联AVFormatContext的流，解码器；out\_buffer、SwsContext、非封装AVFrame大小参数设置的来源

封装AVFrame：接收封装数据，存到非封装AVFrame

非封装AVFrame: 将数据写入非封装文件

音频解码:

音频SwrContext: 关联封装AV\_CH\_LAYOUT\_STEREO, AV\_SAMPLE\_FMT\_S16, 采样率, 非封装AVCodecContext声道布局

音频out\_buffer: 采样率\*2

音频out\_buffer的数据来源: 采样率，封装AVFrame的data和nb\_samples，out\_buffer写入输出文件，没有声明封装AVFrame

音频out\_buffer\_size的数据来源: 声道数(从AV\_CH\_LAYOUT\_STEREO获取)，封装AVFrame的nb\_samples，输出的采样格式

视频编码:

AVFormatContext ->pb: 封装流(输出流)

AVStream: 关联AVFormatContext，后面根据index将AVPacket写入封装文件

编码AVCodecContext: =AVStream->codec codec\_id=AVOutputFormat-> video\_codec codec\_type=AVMEDIA\_TYPE\_VIDEO pix\_fmt、width、height、bit\_rate来自原文件 time\_base.num=1秒 time\_base.den=25帧数 gop\_size=250每250帧插入一个关键帧 qmin = 10 qmax = 51 max\_b\_frames = 0设置b帧的最大值

avcodec\_open2最后一个字典参数：preset预备参数slow, 调优tune零延迟zerolatency

ios: 同时音视频编码 //?lymark android报错: ld: unknown option: -soname

clang: **error:** linker command failed with exit code 1 (use -v to see invocation)

make: \*\*\* [libavutil/libavutil.so.55] Error 1

//lymark为什么不释放: AVStream