# Linux目录结构

# 系统搭建

## **GCC安装**

<https://www.cnblogs.com/tankblog/p/6096681.html>

[linux应用之gcc环境的安装](http://www.cnblogs.com/tankblog/p/6096681.html)

我们知道，用yum install gcc可以安装gcc编译环境，关于 GCC 在 CentOS 下通过 yum 安装默认版本号，CentOS 5 是 4.1.2；CentOS 6 是 4.4.7；CentOS 7 是 4.8.3。

很多时候在编译安装软件都需要高版本的 GCC，否则就会报错。  
那么如何升级 GCC 的版本呢？

首先要确认升级到的 GCC 版本号。  
目前 GCC 的最新版已经到了 5.2，而 CentOS 7 则依然使用其 4.8，所以基于兼容性考虑，我选择升级到 4.8.5。  
GCC 官网：[https://gcc.gnu.org](https://gcc.gnu.org/" \t "https://www.cnblogs.com/tankblog/p/_blank)

下面开始一步一步来编译安装 GCC 4.8.5，需要注意的是，在编译安装 GCC 之前，系统里必须先要通过 yum 安装老版本的 GCC 和依赖库。  
如果是在 x86\_64 系统下编译的话，还需要安装 libgcc.i686 glibc-devel.i686 才行。

yum install -y gcc texinfo-tex flex zip libgcc.i686 glibc-devel.i686

当然，如果是在 CentOS 5 x86\_64 系统下安装，则是如下命令：

yum install -y gcc texinfo-tex flex zip libgcc.i386 glibc-devel.i386

同时需注意，编译安装 GCC 内存不小于 1GB，Swap 不小于 1GB，硬盘最低不小于 10GB，否则极有可能会中途报错退出。  
编译安装完后，目录 gcc-4.8.5 将会有 5GB 之多。

1、下载源码

wget ftp://ftp.gnu.org/gnu/gcc/gcc-4.8.5/gcc-4.8.5.tar.gz

2、下载依赖包  
编译安装 GCC 需要依赖 mpc，mpfr，gmp包。好在 GCC 源码里自带脚本可以轻松下载依赖包。

tar zxf gcc-4.8.5.tar.gz

cd gcc-4.8.5

./contrib/download\_prerequisites

在此脚本里可以看到依赖包的版本号依次是 mpc-0.8.1，mpfr-2.4.2，gmp-4.3.2。

3、编译安装

mkdir gcc-build-4.8.5

cd gcc-build-4.8.5

../configure --prefix=/usr

make && make install

为了避免安装后系统里出现多个版本的 GCC，这里直接将编译安装的目录指定为 /usr，如果不指定 –prefix，则会默认安装到 /usr/local 下。  
GCC 4.8.5 光是源代码就有105MB，因此可以预见整个编译过程需要很长时间（差不多 2 个小时左右）。

4、查看版本号

gcc --version

gcc (GCC) 4.8.5

g++ --version

g++ (GCC) 4.8.5

which gcc

/usr/bin/gcc

which g++

/usr/bin/g++

# 第0部分 关键词

## **0.1 export**

设置程序加载路径

[root@12530test5 bin]# export LD\_LIBRARY\_PATH=/home/ics/search\_serverice/lib/

## **0.2 source**

**理解：**

source命令也称为“点命令”，也就是一个点符号（.）。source命令通常用于重新执行刚修改的初始化文件，使之立即生效，而不必注销并重新登录。

**用法：**

source filename 或 . filename

source命令除了上述的用途之外，还有一个另外一个用途。在对编译系统核心时常常需要输入一长串的命令，如：

make mrproper

make menuconfig

make dep

make clean

make bzImage

…………

如果把这些命令做成一个文件，让它自动顺序执行，对于需要多次反复编译系统核心的用户来说会很方便，而用source命令就可以做到这一点

**备注：**

在修改/etc/profile文件后，期望立即生效可以使用命令{source /etc/profile}

# 第一部分 开发相关

## **1.1 lib/lib64[系统库]**

* 包含支持位于/bin和/sbin下的二进制文件的库文件.
* 库文件名为 ld\*或lib\*.so.\*。例如：ld-2.11.1.so，libncurses.so.5.7
* /lib，/usr/lib，/usr/local/lib：系统使用的函数库的目录，程序在执行过程中，需要调用一些额外的参数时需要函数库的协助，该目录下存放了各种编程语言库。
* /lib目录下的库映像文件可以用来启动系统并执行一些命令，目录/lib/modules包含了可加载的内核模块，/lib目录存放了所有重要的库文件，其他的库文件则大部分存放在/usr/lib目录下。

## **1.2 bin[用户二进制文件 一般用户可用，启动时会用到的命令]**

* 包含二进制可执行文件。
* 在单用户模式下，你需要使用的常见Linux命令都位于此目录下。系统的所有用户使用的命令都设在这里。
* 例如：ps、ls、ping、grep、cp

### **ps**

**样例**

**[root@test25 search]# ps -ef | grep 'palm\_\*'**

UID        PID   PPID  C STIME TTY         TIME  CMD

root 11383 1 0 11:24 pts/0 00:00:00 ./palm\_search\_server palm\_search\_server.conf

root 11493 1 0 Jan25 ? 00:00:43 ./palm\_search\_server palm\_search\_server.conf

root 25803 1 0 Jan30 ? 00:00:05 ./palm\_search\_server palm\_search\_server.conf

root 30544 1 0 Jan30 ? 00:00:07 ./palm\_search\_server palm\_search\_server.conf

-e : 在命令执行后显示环境

-f : 完整显示输出

标为 C 的列是由 CPU 用于计算执行优先级的因子。

 STIME 是指进程的启动时间。

问号表示这些进程不属于任何 TTY，因为它们是由系统启动的。

### **echo**

**理解**

显示环境变量

**样例**

echo $LD\_LIBRARY\_PATH

### **find**

**样例**

Find / -name xxx //在根目录下查找 xxx文件

### **grep**

**理解**

一般跟在管道后边，对前面命令结果过滤

**样例**

netstat -apn | grep ‘LISTEN’

### **uname 查看操作系统版本**

**理解**

查看操作系统信息

**样例**

uname -a

### **ln**

**语法**

ln -s [源文件] [连接文件]

**样例**

//在创建软连接时，可能原来的libc.so.6软连接被干掉了，需要如下第二行才能成功

[root@12530test5 sphinx]# ln -s libc-2.15.so libc.so.6

[root@12530test5 sphinx]# LD\_PRELOAD=/lib64/libc-2.15.so ln -s /lib64/libc-2.15.so /lib64/libc.so.6

## **1.3 sbin[系统二进制文件 开机或单用户模式时还能操作的系统命令]**

* 就像/bin，/sbin同样也包含二进制可执行文件。
* 但是，在这个目录下的linux命令通常由系统管理员使用，对系统进行维护。
* 例如：iptables、reboot、fdisk、ifconfig、swapon命令

### **iptables**

Linux上常用的防火墙软件，是netfilter项目的一部分。可以直接配置，也可以通过许多前端和图形界面配置。

**语法**

iptables(选项)(参数)

**选项**

-t<表>：指定要操纵的表；

-A：向规则链中添加条目；

-D：从规则链中删除条目；

-i：向规则链中插入条目；

-R：替换规则链中的条目；

-L：显示规则链中已有的条目；

-F：清楚规则链中已有的条目；

-Z：清空规则链中的数据包计算器和字节计数器；

-N：创建新的用户自定义规则链；

-P：定义规则链中的默认目标；

-h：显示帮助信息；

-p：指定要匹配的数据包协议类型；

-s：指定要匹配的数据包源ip地址；

-j<目标>：指定要跳转的目标；

-i<网络接口>：指定数据包进入本机的网络接口；

-o<网络接口>：指定数据包要离开本机所使用的网络接口。

**选项输入顺序：**

iptables -t 表名 <-A/I/D/R> 规则链名 [规则号] <-i/o 网卡名> -p 协议名 <-s 源IP/源子网> --sport 源端口 <-d 目标IP/目标子网> --dport 目标端口 -j 动作

表名包括：

raw：高级功能，如：网址过滤。

mangle：数据包修改（QOS），用于实现服务质量。

net：地址转换，用于网关路由器。

filter：包过滤，用于防火墙规则。

规则链名包括：

INPUT链：处理输入数据包。

OUTPUT链：处理输出数据包。

PORWARD链：处理转发数据包。

PREROUTING链：用于目标地址转换（DNAT）。

POSTOUTING链：用于源地址转换（SNAT）。

动作包括：

accept：接收数据包。

DROP：丢弃数据包。

REDIRECT：重定向、映射、透明代理。

SNAT：源地址转换。

DNAT：目标地址转换。

MASQUERADE：IP伪装（NAT），用于ADSL。

LOG：日志记录。

**参数**

## **1.4 usr[系统级的目录 相当于C:/Windows]**

* 包含二进制文件、库文件、文档和二级程序的源代码。
* /usr/bin中包含用户程序的二进制文件。如果你在/bin中找不到用户二进制文件，到/usr/bin目录看看。例如：at、awk、cc、less、scp。
* /usr/sbin中包含系统管理员的二进制文件。如果你在/sbin中找不到系统二进制文件，到/usr/sbin目录看看。例如：atd、cron、sshd、useradd、userdel。
* /usr/lib中包含了/usr/bin和/usr/sbin用到的库。
* /usr/local中包含了从源安装的用户程序。例如，当你从源安装Apache，它会在/usr/local/apache2中。

注意：程序的配置文件、动态的数据文件等都不会存放在/usr，所以除了安装、卸载软件以外，一般无需修改/usr中的内容。在系统正常运行时，/usr甚至可以被只读挂载。于是，/usr常被划分在单独的分区，甚至有时多台计算机可以共享一个/usr。

### **1.4.1 /usr/lib[相当于C:/Windows/Sysytem32]**

系统的库文件。

### **1.4.2 /usr/bin**

多数日常应用程序存放在该目录中。如果/usr被放在单独的分区中，Linux的单用户模式不能访问/usr/bin，所以对系统至关重要的程序不应该放在此文件夹中。

系统用户使用的应用程序。

#### **which**

#### **yum**

**理解**

yum命令是在Fedora和RedHat以及SUSE中基于[rpm](http://man.linuxde.net/rpm" \o "rpm命令" \t "http://man.linuxde.net/_blank)的软件包管理器，它可以使系统管理人员交互和自动化地更细与管理RPM软件包，能够从指定的服务器自动下载RPM包并且安装，可以自动处理依赖性关系，并且一次安装所有依赖的软体包，无须繁琐地一次次下载、安装。

yum提供了查找、安装、删除某一个、一组甚至全部软件包的命令，而且命令简洁而又好记。

**语法**

yum(选项)(参数)

**选项**

-h：显示帮助信息；

-y：对所有的提问都回答“yes”；

-c：指定配置文件；

-q：安静模式；

-v：详细模式；

-d：设置调试等级（0-10）；

-e：设置错误等级（0-10）；

-R：设置yum处理一个命令的最大等待时间；

-C：完全从缓存中运行，而不去下载或者更新任何头文件。

**参数**

install：安装rpm软件包；

update：更新rpm软件包；

check-update：检查是否有可用的更新rpm软件包；

remove：删除指定的rpm软件包；

list：显示软件包的信息；

search：检查软件包的信息；

info：显示指定的rpm软件包的描述信息和概要信息；

clean：清理yum过期的缓存；

shell：进入yum的shell提示符；

resolvedep：显示rpm软件包的依赖关系；

localinstall：安装本地的rpm软件包；

localupdate：显示本地rpm软件包进行更新；

deplist：显示rpm软件包的所有依赖关系。

**样例**

***部分常用的命令包括：***

自动搜索最快镜像插件： yum install yum-fastestmirror

安装yum图形窗口插件： yum install yumex

查看可能批量安装的列表：yum grouplist

***安装***

yum install #全部安装

yum install package1 #安装指定的安装包package1

yum groupinsall group1 #安装程序组group1

***更新和升级***

yum update #全部更新

yum update package1 #更新指定程序包package1

yum check-update #检查可更新的程序

yum upgrade package1 #升级指定程序包package1

yum groupupdate group1 #升级程序组group1

***查找和显示***

yum info package1 #显示安装包信息package1

yum list #显示所有已经安装和可以安装的程序包

yum list package1 #显示指定程序包安装情况package1

yum groupinfo group1 #显示程序组group1信息yum search string 根据关键字string查找安装包

***删除程序***

yum remove &#124; erase package1 #删除程序包package1

yum groupremove group1 #删除程序组group1

yum deplist package1 #查看程序package1依赖情况

***清除缓存***

yum clean packages #清除缓存目录下的软件包

yum clean headers #清除缓存目录下的 headers

yum clean oldheaders #清除缓存目录下旧的 headers

#### **rpm**

#### **find -> ../../bin/find**

find 目录

-ilname<范本样式>：此参数的效果和指定“-lname”参数类似，但忽略字符大小写的差别；

-iname<范本样式>：此参数的效果和指定“-name”参数类似，但忽略字符大小写的差别；

-name<范本样式>：指定字符串作为寻找文件或目录的范本样式

#### **gcc**

#### **gdb**

**实践**

*1、CMakeList中添加调试信息*

ADD\_DEFINITIONS(-Wl,rpath=.:../lib -g -w -fpermissive --std=c++0x -fopenmp -DBIG\_JOINS=1 -fno-strict-aliasing -DLOG\_TIME\_COST )

*2、使用gdb过程*

[root@test25 search]# gdb ./palm\_search\_server //使用gdb挂在程序

gdb set args palm\_search\_server.conf //设置程序参数

gdb b ServerCodec.cpp:250 //设置参数

gdb run //启动程序

gdb n //下一步（直接回车会默认执行上一次的命令）

#### **su**

**理解**:切换用户

#### **sudo**

**理解**

sudo命令用来以其他身份来执行命令，预设的身份为root。在/etc/sudoers中设置了可执行sudo指令的用户。若其未经授权的用户企图使用sudo，则  
会发出警告的邮件给管理员。用户使用sudo时，必须先输入密码，之后有5分钟的有效期限，超过期限则必须重新输入密码。

**语法**

sudo(选项)(参数)

**选项**

-b：在后台执行指令；

-h：显示帮助；

-H：将HOME环境变量设为新身份的HOME环境变量；

-k：结束密码的有效期限，也就是下次再执行sudo时便需要输入密码；。

-l：列出目前用户可执行与无法执行的指令；

-p：改变询问密码的提示符号；

-s<shell>：执行指定的shell；

-u<用户>：以指定的用户作为新的身份。若不加上此参数，则预设以root作为新的身份；

-v：延长密码有效期限5分钟；

-V ：显示版本信息。

**参数**

指令：需要运行的指令和对应的参数。

**实例**

配置sudo必须通过编辑/etc/sudoers文件，而且只有超级用户才可以修改它，还必须使用visudo编辑。之所以使用visudo有两个原因，一是它能够防止两个用户同时修改它；二是它也能进行有限的语法检查。所以，即使只有你一个超级用户，你也最好用visudo来检查一下语法。

visudo默认的是在[vi](http://man.linuxde.net/vi" \o "vi命令" \t "http://man.linuxde.net/_blank)里打开配置文件，用vi来修改文件。我们可以在编译时修改这个默认项。visudo不会擅自保存带有语法错误的配置文件，它会提示你出现的问题，并询问该如何处理，就像：>>> sudoers file: syntax error, line 22 <<

此时我们有三种选择：键入“e”是重新编辑，键入“x”是不保存退出，键入“Q”是退出并保存。如果真选择Q，那么sudo将不会再运行，直到错误被纠正。

etc/sudoers文件格式

目标用户名 主机名=（目标用户名） 命令路径1,命令路径2,....

#### **scp**

**理解**

跨机器拷贝

**样例**

scp hjli11@192.168.86.61:/home/hjli11/SearchEngineDevels/cpp/sichuan\_mobile\_zhangting/DeployBin/bin\_gas/palm\_search\_server palm\_search\_server

scp [builder@192.168.85.218:/home/builder/workspace/CBG.Drip-2ndGroupWMP2-iAvp6.9.6novolte/DBtest/DBTest](mailto:hjli11@192.168.85.218:/home/builder/workspace/CBG.Drip-2ndGroupWMP2-iAvp6.9.6novolte/DBtest/DBtest) DBTest

#### **strings**

显示每个指定的file文件里,包含的所有有4个(或者用选项指定的数字)以上 连续可打印字符的字符串, 在之后紧跟着一个不可打印的字符.默认情况下, 它只显示目标文件初始化和装载段的字符串;对于其它类型的文件它显示整个文件里包含的所有字符串.

[-a|-|--all] [-f|--print-file-name] [-o] [--help] [-v|--version] [-n min-len|-min-len|--bytes=min-len] [-t {o,x,d}[--target=bfdname] |--radix={o,x,d}] file

[root@12530test5 search\_tool]# strings /lib64/libc.so.6 | grep GLIBC

#### **netstat**

netstat -apn //查看网络情况

netstat -apn | grep '12244' | grep 'LISTEN' | awk '{print $7}' | awk -F '/' '{print $1}' //过滤出特定网络程序

#### **awk**

**理解**

按照指定分隔符将文件中的一行行进行处理（注意与grep的区别，grep在于行过滤，awk在于行处理）

**语法**

awk [选项参数] 'script' var=value file(s)  
或  
awk [选项参数] -f scriptfile var=value file(s)

**选项参数说明：**

-F fs or --field-separator fs  
指定输入文件折分隔符，fs是一个字符串或者是一个正则表达式，如-F:。  
-v var=value or --asign var=value  
赋值一个用户定义变量。  
-f scripfile or --file scriptfile  
从脚本文件中读取awk命令。  
-mf nnn and -mr nnn  
对nnn值设置内在限制，-mf选项限制分配给nnn的最大块数目；-mr选项限制记录的最大数目。这两个功能是Bell实验室版awk的扩展功能，在标准awk中不适用。  
-W compact or --compat, -W traditional or --traditional  
在兼容模式下运行awk。所以gawk的行为和标准的awk完全一样，所有的awk扩展都被忽略。  
-W copyleft or --copyleft, -W copyright or --copyright  
打印简短的版权信息。  
-W help or --help, -W usage or --usage  
打印全部awk选项和每个选项的简短说明。  
-W lint or --lint  
打印不能向传统unix平台移植的结构的警告。  
-W lint-old or --lint-old  
打印关于不能向传统unix平台移植的结构的警告。  
-W posix  
打开兼容模式。但有以下限制，不识别：/x、函数关键字、func、换码序列以及当fs是一个空格时，将新行作为一个域分隔符；操作符\*\*和\*\*=不能代替^和^=；fflush无效。  
-W re-interval or --re-inerval  
允许间隔正则表达式的使用，参考(grep中的Posix字符类)，如括号表达式[[:alpha:]]。  
-W source program-text or --source program-text  
使用program-text作为源代码，可与-f命令混用。  
-W version or --version  
打印bug报告信息的版本。

#### **zip/unzip**

1，当前目录

zip -r xxx.zip ./\*             //压缩服务器上当前目录的内容为xxx.zip文件

unzip filename.zip        //解压zip文件到当前目录

2，命令说明

zip -r myfile.zip ./\*

将当前目录下的所有文件和文件夹全部压缩成myfile.zip文件,－r表示递归压缩子目录下所有文件.

unzip -o -d /home/sunny myfile.zip

把myfile.zip文件解压到 /home/sunny/

-o:不提示的情况下覆盖文件；

-d:-d /home/sunny 指明将文件解压缩到/home/sunny目录下；

3.其他

zip -d myfile.zip smart.txt               //删除压缩文件中smart.txt文件

zip -m myfile.zip ./rpm\_info.txt       //向压缩文件中myfile.zip中添加rpm\_info.txt文件

#### **tar**

#### **wget**

### **1.4.3 /usr/sbin**

在单用户模式中不用的系统管理程序，多数软件安装在此。

超级用户使用的比较高级的管理程序和系统守护程序。

#### **visudo**

专门用于编辑sudo命令的配置文件/etc/sudoers

### **1.4.4 /usr/include**

存放C/C++头文件的目录。

### **1.4.5 /usr/src[系统级的源码]**

存放源代码。

### **1.4.6 /usr/share[放置共享文件的地方]**

#### 1.4.6.1 /usr/share/man**[联机帮助文件]**

#### 1.4.6.2 /usr/share/doc**[软件杂项的文件说明]**

#### 1.4.6.3 /usr/share/zoneinfo**[与时区有关的时区档案]**

### **1.4.7 /usr/man /usr/info /usr/doc**

手册页、GNU信息文档和各种其他文档文件

### **1.4.8 /usr/local[=C:/Progrem Files 自己编译的软件默认安装目录]**

* 在新装的系统中这个文件夹是空的，可以用于存放个人安装的软件。安装了本地软件的/usr/local里的目录结构与/usr相似。
* 这里主要存放那些手动安装的软件，即不是通过“新立得”或apt-get安装的软件。它和/usr目录具有相类似的目录结构。  
  让软件包管理器来管理/usr目录，而把自定义的脚本(scripts)放到/usr/local目录下面，我想这应该是个不错的主意。

ps:新立得（Synaptic）是Debian和Ubuntu Linux操作系统的包管理工具apt的图形化前端。它结合了图形界面的简单操作和 apt-get 命令行工具的强大功能

#### 1.4.8.1 /usr/local/src**[用户级的源码]**

#### 1.4.8.2 /usr/local/bin

##### cmake

###### 安装

1、从官网下载cmake安装包，输入指令解压缩

sudo tar -zxvf cmake-3.8.2.tar.gz

2、进入解压缩后的文件夹中，依次输入

sudo ./bootstrap

sudo make

sudo make install

3、安装检查：输入命令，

cmake --version

若出现对应cmake的版本，则说明安装成功，一般默认安装在 /usr/local/bin 目录下。

###### 简单样例

a、新建目录hello, 依次创建文件main.c, CMakeLists.txt, build目录

b、main.c 文件代码如下

#include <stdio.h>

int main(void){

printf("Hello\n");

return 0;

}

c、CMakeLists.txt 文件代码如下

CMAKE\_MINIMUM\_REQUIRED(VERSION 3.8) #cmake最低版本需求，不加入此行会受到警告信息

PROJECT(HELLO) #项目名称

AUX\_SOURCE\_DIRECTORY(. SRC\_LIST) #把当前目录(.)下所有源代码文件和头文件加入变量SRC\_LIST

ADD\_EXECUTABLE(hello ${SRC\_LIST}) #生成应用程序 hello (在windows下会自动生成hello.exe)

d、进入build 目录，依次输入

cmake ..

make

./hello

即可得到相应的结果

###### 复杂样例

hjli11@ms01:~/build\_dir$ cmake -DCMAKE\_BUILD\_TYPE=RELEASE -DCXX\_SERVER\_OPTION=palm ..

hjli11@ms01:~/build\_dir$ make -j 12

|  |  |
| --- | --- |
| *集成环境编译预设变量*  set(**GCC\_COMPLER\_ROOT**  "/home/cis/gcc4.8.5")  set(**boost\_lib\_path** "/home/cis/boost\_gcc\_lib")  set(**CMAKE\_C\_COMPILER** ${**GCC\_COMPLER\_ROOT**}/bin/gcc)  set(**CMAKE\_CXX\_COMPILER** ${**GCC\_COMPLER\_ROOT**}/bin/g++)  set(**LIB\_STDCPLUS** ${**GCC\_COMPLER\_ROOT**}/lib64)  set(**STDFLAG** "-std=c++11") | cmake/cmake.ci  设置变量**GCC\_COMPLER\_ROOT**  设置变量 |
| *默认环境编译预设变量*  #set(**GCC\_COMPLER\_ROOT** "/home/taoli5/gcc4.8.5/gcc4.8.5\_locate")  set(**boost\_lib\_path** "/usr/local/lib/boost\_gcc\_lib")  #set(CMAKE\_C\_COMPILER ${GCC\_COMPLER\_ROOT}/bin/gcc)  #set(CMAKE\_CXX\_COMPILER ${GCC\_COMPLER\_ROOT}/bin/g++)  #set(LIB\_STDCPLUS ${GCC\_COMPLER\_ROOT}/lib64)  set (**STDFLAG** "-std=c++0x") | cmake/cmake.default |
| *开发环境预设变量*  set(**GCC\_COMPLER\_ROOT** "/home/taoli5/gcc4.8.5/gcc4.8.5\_locate")  set(**boost\_lib\_path** "/usr/local/lib/boost\_gcc\_lib")  set(**CMAKE\_C\_COMPILER** ${**GCC\_COMPLER\_ROOT**}/bin/gcc)  set(**CMAKE\_CXX\_COMPILER** ${**GCC\_COMPLER\_ROOT**}/bin/g++)  set(**LIB\_STDCPLUS** ${**GCC\_COMPLER\_ROOT**}/lib64)  set(**STDFLAG** "-std=c++11") | cmake/cmake.develop |
| **编译环境判断**  if (**BUILD\_ENV** STREQUAL "Dev")  include(./cmake/cmake.develop)  elseif (**BUILD\_ENV** STREQUAL "Ci")  include(./cmake/cmake.ci)  else (true)  include(./cmake/cmake.default)  endif()  **编译器的选择**  IF (**CXX\_COMPILER\_OPTION** STREQUAL "Intel")  MESSAGE(STATUS "\*\* Compiling with Intel settings \*\*")  set(**CMAKE\_C\_COMPILER** "/opt/intel/bin/icc")  set(**CMAKE\_CXX\_COMPILER** "/opt/intel/bin/icpc")  SET(**THRD\_LIB\_PATH**  ${**boost\_lib\_path**} /opt/intel/lib/intel64)  SET(**OMP\_LIB** iomp5)  SET(**ORTHER\_DEFINITIONS** -pthread -openmp -vec -parallel)  **ELSEIF** (**CXX\_COMPILER\_OPTION** STREQUAL "48")  MESSAGE(STATUS "\*\* Compiling with 48 settings \*\*")  set(**CMAKE\_C\_COMPILER** "/home/taoli5/gcc4.8.5/gcc4.8.5\_locate/bin/gcc")  set(**CMAKE\_CXX\_COMPILER** "/home/taoli5/gcc4.8.5/gcc4.8.5\_locate/bin/c++")  SET(**THRD\_LIB\_PATH** /usr/local/lib/boost\_gcc\_lib)  SET(**OMP\_LIB** gomp)  SET(**ORTHER\_DEFINITIONS** -pthread)  **ELSE** (true)  SET(**THRD\_LIB\_PATH** ${**boost\_lib\_path**} ${**LIB\_STDCPLUS**})  SET(**OMP\_LIB** gomp)  SET(**ORTHER\_DEFINITIONS** -pthread)  **ENDIF**()  **是否启用监视**  IF (**CXX\_KNOWLEDGE\_CLOUD\_MONITOER**)  SET(**ACE\_LIB** ACE)  ELSE (true)  SET(**ACE\_LIB** )  ENDIF()  **PROJECT**(music\_search)  **MESSAGE**("---------------音乐搜索构建------------------")  SET ( **TARGETDIR** ${**PROJECT\_SOURCE\_DIR**}/DeployBin )  SET ( **ROOT\_DIR** ${**PROJECT\_SOURCE\_DIR**} )  **MESSAGE** ( "生成目标文件夹 :${**TARGETDIR**}" )  #set (**STDFLAG** "-std=c++11")  **CMAKE\_MINIMUM\_REQUIRED**(VERSION 2.8)  **导入依赖项目**  MESSAGE("导入：music\_baselib")  ADD\_SUBDIRECTORY(music\_baselib)  MESSAGE("导入：music\_asiomodel")  ADD\_SUBDIRECTORY(music\_asiomodel)  MESSAGE("导入：music\_segmentor")  ADD\_SUBDIRECTORY(music\_segmentor)  #MESSAGE("导入：music\_modulelib")  #ADD\_SUBDIRECTORY(music\_modulelib)  #MESSAGE("导入：music\_sentsimlib")  #ADD\_SUBDIRECTORY(music\_sentsimlib)  #MESSAGE("导入：music\_intent")  #ADD\_SUBDIRECTORY(music\_intent)  #MESSAGE("导入：music\_intent")  #ADD\_SUBDIRECTORY(music\_intent)  #MESSAGE("导入:music\_preproc")  #ADD\_SUBDIRECTORY(music\_preproc)  #MESSAGE("导入 :music\_preproc\_new")  #ADD\_SUBDIRECTORY(music\_preproc\_new)  **编译指定项目**  IF (**CXX\_SERVER\_OPTION** STREQUAL "engine")  MESSAGE("导入：music\_searchengine")  ADD\_SUBDIRECTORY(music\_searchengine)  ELSEIF (**CXX\_SERVER\_OPTION** STREQUAL "migu")  MESSAGE("导入：music\_searchengine")  ADD\_SUBDIRECTORY(music\_migusearch)  ELSEIF(**CXX\_SERVER\_OPTION** STREQUAL "kuyin")  MESSAGE("导入:music\_searchengine\_kuyin")  ADD\_SUBDIRECTORY(music\_searchengine\_kuyin)  ELSEIF(**CXX\_SERVER\_OPTION** STREQUAL "assoc")  MESSAGE("导入:music\_assoc\_new")  ADD\_SUBDIRECTORY(music\_assoc\_new)  ELSEIF(CXX\_SERVER\_OPTION STREQUAL "aas")  MESSAGE("导入:audio\_search\_aas")  ADD\_SUBDIRECTORY(AudioSearch)  ELSEIF(**CXX\_SERVER\_OPTION** STREQUAL "afs")  MESSAGE("导入:audio\_search\_afs")  ADD\_SUBDIRECTORY(AFSearch)  ELSEIF(**CXX\_SERVER\_OPTION** STREQUAL "story")  MESSAGE("导入:story\_searchengine")  ADD\_SUBDIRECTORY(story\_searchengine)  ELSEIF(**CXX\_SERVER\_OPTION** STREQUAL "sand")  MESSAGE("导入:sang\_searchengine")  ADD\_SUBDIRECTORY(sand\_searchengine)  ELSEIF(**CXX\_SERVER\_OPTION** STREQUAL "palm")  MESSAGE("导入:palm\_searchengine")  ADD\_SUBDIRECTORY(palm\_searchengine)  ELSEIF(**CXX\_SERVER\_OPTION** STREQUAL "wotv")  MESSAGE("导入:wotv\_searchengine")  ADD\_SUBDIRECTORY(wotv\_searchengine)  ELSEIF(**CXX\_SERVER\_OPTION** STREQUAL "aiui")  MESSAGE("导入:aiui\_searchengine")  ADD\_SUBDIRECTORY(music\_searchengine\_aiui)  ELSEIF(**CXX\_SERVER\_OPTION** STREQUAL "iptv")  MESSAGE("导入:iptv\_searchengine")  ADD\_SUBDIRECTORY(iptv\_searchengine)  ELSEIF(**CXX\_SERVER\_OPTION** STREQUAL "mixtoler")  MESSAGE("导入:mix\_toler")  ADD\_SUBDIRECTORY(mix\_toler)  ELSEIF(**CXX\_SERVER\_OPTION** STREQUAL "toler")  MESSAGE("导入：Music\_Toler\_New")  ADD\_SUBDIRECTORY(music\_toler\_new)  MESSAGE("导入：ResourceBuilders")  ADD\_SUBDIRECTORY(music\_toler\_new/ACFTAssoResourceBuild)  ELSE (true)  MESSAGE("导入：music\_searchengine")  ADD\_SUBDIRECTORY(music\_searchengine)  MESSAGE("导入:music\_assoc\_new")  ADD\_SUBDIRECTORY(music\_assoc\_new)  MESSAGE("导入：Music\_Toler\_New")  ADD\_SUBDIRECTORY(music\_toler\_new)  MESSAGE("导入：ResourceBuilders")  ADD\_SUBDIRECTORY(music\_toler\_new/ACFTAssoResourceBuild)  ENDIF()  ADD\_DEFINITIONS(-w -Wno-unused-result ) | 使用**BUILD\_ENV**设置编译环境  设置C编译器环境路径  设置C++编译器环境路径 |
| project(palm\_search\_server)  CMAKE\_MINIMUM\_REQUIRED(VERSION 2.8)  AUX\_SOURCE\_DIRECTORY( ./ DIR\_SRCS)  AUX\_SOURCE\_DIRECTORY(../music\_3rdref/libsphinxclient2.1.3/src DIR\_SRCS\_SPC)  message( "CMAKE\_CURRENT\_SOURCE\_DIR ${CMAKE\_CURRENT\_SOURCE\_DIR}")  message( "TARGETDIR ${TARGETDIR}")  INCLUDE\_DIRECTORIES( ./  ../music\_3rdref/libsphinxclient2.1.3/include  ../music\_3rdref/libsphinxclient2.1.3/src  ../music\_asiomodel  ../music\_baselib  ../music\_baselib/logs  ../music\_3rdref/protobuff/include  ../music\_3rdref/intent/include  ../music\_redis/deps/hiredis  ../music\_intent  ../music\_intent/include  ../music\_baselib/pmi  /usr/include/mysql)  SET(LIB\_SRC ${DIR\_SRCS} ${DIR\_SRCS\_SPC} )  LINK\_DIRECTORIES( ${TARGETDIR}/lib ${TARGETDIR}/lib/pmi\_lib ${TARGETDIR}/bin\_gas ${THRD\_LIB\_PATH})  ADD\_DEFINITIONS(-Wl,rpath=.:../lib -g -w -fpermissive --std=c++0x -fopenmp -DBIG\_JOINS=1 -fno-strict-aliasing -DLOG\_TIME\_COST )  IF(CXX\_ENGINE\_VERSION)  ADD\_DEFINITIONS(-DENGINE\_VERSION="${CXX\_ENGINE\_VERSION}")  ENDIF()  IF(CXX\_KNOWLEDGE\_CLOUD\_MONITOER)  ADD\_DEFINITIONS(-DKNOWLEDGE\_CLOUD\_MONITOER="${CXX\_KNOWLEDGE\_CLOUD\_MONITOER}")  ENDIF()  SET(CMAKE\_CXX\_FLAGS\_DEBUG "$ENV{CXXFLAGS} -g ")  SET(CMAKE\_CXX\_FLAGS\_RELEASE "$ENV{CXXFLAGS} -O3 -g")  SET(CMAKE\_EXE\_LINKER\_FLAGS "-fopenmp" )  SET(CMAKE\_EXE\_LINKER\_FLAGS "-Wl,--copy-dt-needed-entries")  SET(EXECUTABLE\_OUTPUT\_PATH ${TARGETDIR}/bin\_gas)  add\_executable( palm\_search\_server ${LIB\_SRC} )  FIND\_LIBRARY(SLibrarys libboost\_regex.a libboost\_system.a libboost\_thread.a libboost\_filesystem.a libboost\_date\_time.a  PATHS ${THRD\_LIB\_PATH})  ADD\_DEFINITIONS( ${ORTHER\_DEFINITIONS})  IF (SLibrarys)  TARGET\_LINK\_LIBRARIES(palm\_search\_server LINK\_PRIVATE segmentor AsioModel BaseLib protobuf libboost\_regex.a libboost\_system.a libboost\_thread.a libboost\_filesystem.a libboost\_date\_time.a hiredis mysqlclient m rt dl ${OMP\_LIB} ${ACE\_LIB})  ELSE(true)  TARGET\_LINK\_LIBRARIES(music\_search\_server LINK\_PRIVATE segmentor AsioModel BaseLib protobuf boost\_regex pthread boost\_system boost\_thread boost\_filesystem boost\_date\_time hiredis mysqlclient m rt dl ${OMP\_LIB} ${ACE\_LIB})  ENDIF(SLibrarys) | **O**1 在不影响编译速度的前提下，尽量采用一些优化算法降低代码大小和可执行代码的运行速度  **O**2 该优化选项会牺牲部分编译速度，除了执行-O1所执行的所有优化之外，还会采用几乎所有的目标配置支持的优化算法，用以提高目标代码的运行速度。  O3 该选项除了执行-O2所有的优化选项之外，一般都是采取很多向量化算法，提高代码的并行执行程度，利用现代CPU中的流水线，Cache等。  **Os** 这个优化标识和-O3有异曲同工之妙，当然两者的目标不一样，-O3的目标是宁愿增加目标代码的大小，也要拼命的提高运行速度，但是这个选项是在-O2的基础之上，尽量的降低目标代码的大小，这对于存储容量很小的设备来说非常重要。  **Ofast** 该选项将不会严格遵循语言标准，除了启用所有的-O3优化选项之外，也会针对某些语言启用部分优化  **Og** 该标识会精心挑选部分与-g选项不冲突的优化选项，当然就能提供合理的优化水平，同时产生较好的可调试信息和对语言标准的遵循程度 |

###### 简介

cmake  [ < 选项> ]  （< path - to - source >  |  < path - to - existing - build > ）  
cmake  [（- D  < var > = < value > ）... ]  - P  < cmake - script - file >   
cmake  - build  < dir >  [ < 选项> [ ]  < - build - tool - options > ... ] cmake - E < command > [ < options > ... ] cmake - find - package < options > ...

###### 说明

“cmake”可执行文件是CMake命令行界面。它可以用来在脚本中配置项目。可以使用-D选项在命令行上指定项目配置设置。

CMake是一个跨平台的构建系统生成器。项目使用名为CMakeLists.txt的源代码树的每个目录中包含与平台无关的CMake列表文件来指定其构建过程。用户通过使用CMake为其平台上的本地工具生成构建系统来构建项目。

###### 选项

###### Variables that Provide Information

* [CMAKE\_ARGC](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ARGC.html)

在脚本模式下传递给CMake的命令行参数的数量。

* [CMAKE\_ARGV0](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ARGV0.html)

命令航参数一脚本模式传递给CMake

* [CMAKE\_AR](https://cmake.org/cmake/help/v3.5/variable/CMAKE_AR.html)
* [CMAKE\_BINARY\_DIR](https://cmake.org/cmake/help/v3.5/variable/CMAKE_BINARY_DIR.html)
* [CMAKE\_BUILD\_TOOL](https://cmake.org/cmake/help/v3.5/variable/CMAKE_BUILD_TOOL.html)
* [CMAKE\_CACHEFILE\_DIR](https://cmake.org/cmake/help/v3.5/variable/CMAKE_CACHEFILE_DIR.html)
* [CMAKE\_CACHE\_MAJOR\_VERSION](https://cmake.org/cmake/help/v3.5/variable/CMAKE_CACHE_MAJOR_VERSION.html)
* [CMAKE\_CACHE\_MINOR\_VERSION](https://cmake.org/cmake/help/v3.5/variable/CMAKE_CACHE_MINOR_VERSION.html)
* [CMAKE\_CACHE\_PATCH\_VERSION](https://cmake.org/cmake/help/v3.5/variable/CMAKE_CACHE_PATCH_VERSION.html)
* [CMAKE\_CFG\_INTDIR](https://cmake.org/cmake/help/v3.5/variable/CMAKE_CFG_INTDIR.html)
* [CMAKE\_COMMAND](https://cmake.org/cmake/help/v3.5/variable/CMAKE_COMMAND.html)

cmake命令的完整路径

* [CMAKE\_CROSSCOMPILING](https://cmake.org/cmake/help/v3.5/variable/CMAKE_CROSSCOMPILING.html)

如果CMake是交叉编译的话，这个变量将被设置为真。具体而言，如果构建平台与目标平台不同。

* [CMAKE\_CROSSCOMPILING\_EMULATOR](https://cmake.org/cmake/help/v3.5/variable/CMAKE_CROSSCOMPILING_EMULATOR.html)
* [CMAKE\_CTEST\_COMMAND](https://cmake.org/cmake/help/v3.5/variable/CMAKE_CTEST_COMMAND.html)

这是CTest可执行ctest的完整路径，这对于想要使用cmake -E选项进行可移植系统命令的自定义命令很有用。

* [CMAKE\_CURRENT\_BINARY\_DIR](https://cmake.org/cmake/help/v3.5/variable/CMAKE_CURRENT_BINARY_DIR.html)

这是cmake正在处理的构建目录的完整路径。由add\_subdirectory添加的每个目录将在构建树中创建一个二进制目录，并且在处理该变量时将被设置。对于源内部版本，这是当前正在处理的源目录。

* [CMAKE\_CURRENT\_LIST\_DIR](https://cmake.org/cmake/help/v3.5/variable/CMAKE_CURRENT_LIST_DIR.html)

当前正在处理的列表文件的完整目录。

当CMake处理项目中的列表文件时，此变量将始终设置为当前正在处理的列表文件（CMAKE\_CURRENT\_LIST\_FILE）所在的目录。该值具有动态范围。当CMake开始处理源文件中的命令时，它将此变量设置为该文件所在的目录。当CMake完成处理来自文件的命令时，它恢复以前的值。因此，宏或函数中变量的值是调用调用堆栈最底层条目的文件的目录，而不是包含宏或函数定义的文件的目录。

* [CMAKE\_CURRENT\_LIST\_FILE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_CURRENT_LIST_FILE.html)

当前正在处理的列表文件的完整路径。

当CMake处理项目中的列表文件时，这个变量总是被设置为当前正在处理的那个。该值具有动态范围。当CMake开始处理源文件中的命令时，它将这个变量设置为文件的位置。当CMake完成处理来自文件的命令时，它恢复以前的值。因此，宏或函数中变量的值是调用调用堆栈中最底层条目的文件，而不是包含宏或函数定义的文件。

* [CMAKE\_CURRENT\_LIST\_LINE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_CURRENT_LIST_LINE.html)

这是cmake当前正在处理的文件的行号。

* [CMAKE\_CURRENT\_SOURCE\_DIR](https://cmake.org/cmake/help/v3.5/variable/CMAKE_CURRENT_SOURCE_DIR.html)

当前处理的CMakeLists.txt所在的路径

这是cmake正在处理的源目录的完整路径。

* [CMAKE\_DL\_LIBS](https://cmake.org/cmake/help/v3.5/variable/CMAKE_DL_LIBS.html)

dlopen和dlclose的库的名称，在大多数UNIX机器上通常是-ldl。

* [CMAKE\_EDIT\_COMMAND](https://cmake.org/cmake/help/v3.5/variable/CMAKE_EDIT_COMMAND.html)

cmake-gui或ccmake的完整路径。当不使用IDE的“额外”生成器时，仅为Makefile生成器定义。

这是可以图形化编辑缓存的CMake可执行文件的完整路径。例如，cmake-gui或ccmake。

* [CMAKE\_EXECUTABLE\_SUFFIX](https://cmake.org/cmake/help/v3.5/variable/CMAKE_EXECUTABLE_SUFFIX.html)

此平台上可执行文件的后缀。

在Windows上使用可执行文件名后缀（如果有的话）的后缀。

CMAKE\_EXECUTABLE\_SUFFIX\_ <LANG>将替代语言<LANG>。

* [CMAKE\_EXTRA\_GENERATOR](https://cmake.org/cmake/help/v3.5/variable/CMAKE_EXTRA_GENERATOR.html)

用于构建项目的额外的生成器。

当使用Eclipse，CodeBlocks或KDevelop生成器时，CMake生成Makefile（CMAKE\_GENERATOR），并为各自的IDE生成额外的项目文件。此IDE项目文件生成器存储在CMAKE\_EXTRA\_GENERATOR（例如“Eclipse CDT4”）中。

* [CMAKE\_EXTRA\_SHARED\_LIBRARY\_SUFFIXES](https://cmake.org/cmake/help/v3.5/variable/CMAKE_EXTRA_SHARED_LIBRARY_SUFFIXES.html)

额外的共享库后缀。

除CMAKE\_SHARED\_LIBRARY\_SUFFIX指定的共享库之外的扩展（如果有的话）。CMake在分析由目标链接的库时使用它来识别外部共享库文件。

* [CMAKE\_FIND\_PACKAGE\_NAME](https://cmake.org/cmake/help/v3.5/variable/CMAKE_FIND_PACKAGE_NAME.html)
* [CMAKE\_GENERATOR](https://cmake.org/cmake/help/v3.5/variable/CMAKE_GENERATOR.html)

生成器用于构建项目。

正在用来生成构建文件的生成器的名称。（例如“Unix Makefiles”，“Visual Studio 6”等）

* [CMAKE\_GENERATOR\_PLATFORM](https://cmake.org/cmake/help/v3.5/variable/CMAKE_GENERATOR_PLATFORM.html)
* [CMAKE\_GENERATOR\_TOOLSET](https://cmake.org/cmake/help/v3.5/variable/CMAKE_GENERATOR_TOOLSET.html)

由用户指定的本机构建系统工具集名称。

一些CMake生成器支持一个工具集名称，以赋予本地构建系统选择一个编译器。如果用户指定一个工具集名称（例如，通过cmake -T选项），该值将在这个变量中可用。

* [CMAKE\_HOME\_DIRECTORY](https://cmake.org/cmake/help/v3.5/variable/CMAKE_HOME_DIRECTORY.html)
* [CMAKE\_IMPORT\_LIBRARY\_PREFIX](https://cmake.org/cmake/help/v3.5/variable/CMAKE_IMPORT_LIBRARY_PREFIX.html)
* [CMAKE\_IMPORT\_LIBRARY\_SUFFIX](https://cmake.org/cmake/help/v3.5/variable/CMAKE_IMPORT_LIBRARY_SUFFIX.html)
* [CMAKE\_JOB\_POOL\_COMPILE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_JOB_POOL_COMPILE.html)
* [CMAKE\_JOB\_POOL\_LINK](https://cmake.org/cmake/help/v3.5/variable/CMAKE_JOB_POOL_LINK.html)
* [CMAKE\_LINK\_LIBRARY\_SUFFIX](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LINK_LIBRARY_SUFFIX.html)
* [CMAKE\_LINK\_SEARCH\_END\_STATIC](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LINK_SEARCH_END_STATIC.html)
* [CMAKE\_LINK\_SEARCH\_START\_STATIC](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LINK_SEARCH_START_STATIC.html)
* [CMAKE\_MAJOR\_VERSION](https://cmake.org/cmake/help/v3.5/variable/CMAKE_MAJOR_VERSION.html)
* [CMAKE\_MAKE\_PROGRAM](https://cmake.org/cmake/help/v3.5/variable/CMAKE_MAKE_PROGRAM.html)
* [CMAKE\_MATCH\_COUNT](https://cmake.org/cmake/help/v3.5/variable/CMAKE_MATCH_COUNT.html)
* [CMAKE\_MINIMUM\_REQUIRED\_VERSION](https://cmake.org/cmake/help/v3.5/variable/CMAKE_MINIMUM_REQUIRED_VERSION.html)
* [CMAKE\_MINOR\_VERSION](https://cmake.org/cmake/help/v3.5/variable/CMAKE_MINOR_VERSION.html)
* [CMAKE\_PARENT\_LIST\_FILE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_PARENT_LIST_FILE.html)
* [CMAKE\_PATCH\_VERSION](https://cmake.org/cmake/help/v3.5/variable/CMAKE_PATCH_VERSION.html)
* [CMAKE\_PROJECT\_NAME](https://cmake.org/cmake/help/v3.5/variable/CMAKE_PROJECT_NAME.html)
* [CMAKE\_RANLIB](https://cmake.org/cmake/help/v3.5/variable/CMAKE_RANLIB.html)
* [CMAKE\_ROOT](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ROOT.html)
* [CMAKE\_SCRIPT\_MODE\_FILE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SCRIPT_MODE_FILE.html)
* [CMAKE\_SHARED\_LIBRARY\_PREFIX](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SHARED_LIBRARY_PREFIX.html)
* [CMAKE\_SHARED\_LIBRARY\_SUFFIX](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SHARED_LIBRARY_SUFFIX.html)
* [CMAKE\_SHARED\_MODULE\_PREFIX](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SHARED_MODULE_PREFIX.html)
* [CMAKE\_SHARED\_MODULE\_SUFFIX](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SHARED_MODULE_SUFFIX.html)
* [CMAKE\_SIZEOF\_VOID\_P](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SIZEOF_VOID_P.html)
* [CMAKE\_SKIP\_INSTALL\_RULES](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SKIP_INSTALL_RULES.html)
* [CMAKE\_SKIP\_RPATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SKIP_RPATH.html)
* [CMAKE\_SOURCE\_DIR](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SOURCE_DIR.html)
* [CMAKE\_STANDARD\_LIBRARIES](https://cmake.org/cmake/help/v3.5/variable/CMAKE_STANDARD_LIBRARIES.html)
* [CMAKE\_STATIC\_LIBRARY\_PREFIX](https://cmake.org/cmake/help/v3.5/variable/CMAKE_STATIC_LIBRARY_PREFIX.html)
* [CMAKE\_STATIC\_LIBRARY\_SUFFIX](https://cmake.org/cmake/help/v3.5/variable/CMAKE_STATIC_LIBRARY_SUFFIX.html)
* [CMAKE\_TOOLCHAIN\_FILE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_TOOLCHAIN_FILE.html)
* [CMAKE\_TWEAK\_VERSION](https://cmake.org/cmake/help/v3.5/variable/CMAKE_TWEAK_VERSION.html)
* [CMAKE\_VERBOSE\_MAKEFILE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_VERBOSE_MAKEFILE.html)
* [CMAKE\_VERSION](https://cmake.org/cmake/help/v3.5/variable/CMAKE_VERSION.html)
* [CMAKE\_VS\_DEVENV\_COMMAND](https://cmake.org/cmake/help/v3.5/variable/CMAKE_VS_DEVENV_COMMAND.html)
* [CMAKE\_VS\_INTEL\_Fortran\_PROJECT\_VERSION](https://cmake.org/cmake/help/v3.5/variable/CMAKE_VS_INTEL_Fortran_PROJECT_VERSION.html)
* [CMAKE\_VS\_MSBUILD\_COMMAND](https://cmake.org/cmake/help/v3.5/variable/CMAKE_VS_MSBUILD_COMMAND.html)
* [CMAKE\_VS\_MSDEV\_COMMAND](https://cmake.org/cmake/help/v3.5/variable/CMAKE_VS_MSDEV_COMMAND.html)
* [CMAKE\_VS\_NsightTegra\_VERSION](https://cmake.org/cmake/help/v3.5/variable/CMAKE_VS_NsightTegra_VERSION.html)
* [CMAKE\_VS\_PLATFORM\_NAME](https://cmake.org/cmake/help/v3.5/variable/CMAKE_VS_PLATFORM_NAME.html)
* [CMAKE\_VS\_PLATFORM\_TOOLSET](https://cmake.org/cmake/help/v3.5/variable/CMAKE_VS_PLATFORM_TOOLSET.html)
* [CMAKE\_VS\_WINDOWS\_TARGET\_PLATFORM\_VERSION](https://cmake.org/cmake/help/v3.5/variable/CMAKE_VS_WINDOWS_TARGET_PLATFORM_VERSION.html)
* [CMAKE\_XCODE\_PLATFORM\_TOOLSET](https://cmake.org/cmake/help/v3.5/variable/CMAKE_XCODE_PLATFORM_TOOLSET.html)
* [PROJECT\_BINARY\_DIR](https://cmake.org/cmake/help/v3.5/variable/PROJECT_BINARY_DIR.html) 运行cmake命令的目录

通常是${PROJECT\_SOURCE\_DIR}/build

* [<PROJECT-NAME>\_BINARY\_DIR](https://cmake.org/cmake/help/v3.5/variable/PROJECT-NAME_BINARY_DIR.html)
* [PROJECT\_NAME](https://cmake.org/cmake/help/v3.5/variable/PROJECT_NAME.html)
* [<PROJECT-NAME>\_SOURCE\_DIR](https://cmake.org/cmake/help/v3.5/variable/PROJECT-NAME_SOURCE_DIR.html)
* [<PROJECT-NAME>\_VERSION](https://cmake.org/cmake/help/v3.5/variable/PROJECT-NAME_VERSION.html)
* [<PROJECT-NAME>\_VERSION\_MAJOR](https://cmake.org/cmake/help/v3.5/variable/PROJECT-NAME_VERSION_MAJOR.html)
* [<PROJECT-NAME>\_VERSION\_MINOR](https://cmake.org/cmake/help/v3.5/variable/PROJECT-NAME_VERSION_MINOR.html)
* [<PROJECT-NAME>\_VERSION\_PATCH](https://cmake.org/cmake/help/v3.5/variable/PROJECT-NAME_VERSION_PATCH.html)
* [<PROJECT-NAME>\_VERSION\_TWEAK](https://cmake.org/cmake/help/v3.5/variable/PROJECT-NAME_VERSION_TWEAK.html)
* [PROJECT\_SOURCE\_DIR](https://cmake.org/cmake/help/v3.5/variable/PROJECT_SOURCE_DIR.html) 工程根目录
* [PROJECT\_VERSION](https://cmake.org/cmake/help/v3.5/variable/PROJECT_VERSION.html)
* [PROJECT\_VERSION\_MAJOR](https://cmake.org/cmake/help/v3.5/variable/PROJECT_VERSION_MAJOR.html)
* [PROJECT\_VERSION\_MINOR](https://cmake.org/cmake/help/v3.5/variable/PROJECT_VERSION_MINOR.html)
* [PROJECT\_VERSION\_PATCH](https://cmake.org/cmake/help/v3.5/variable/PROJECT_VERSION_PATCH.html)
* [PROJECT\_VERSION\_TWEAK](https://cmake.org/cmake/help/v3.5/variable/PROJECT_VERSION_TWEAK.html)

###### Variables that Change Behavior

* [BUILD\_SHARED\_LIBS](https://cmake.org/cmake/help/v3.5/variable/BUILD_SHARED_LIBS.html)
* [CMAKE\_ABSOLUTE\_DESTINATION\_FILES](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ABSOLUTE_DESTINATION_FILES.html)
* [CMAKE\_APPBUNDLE\_PATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_APPBUNDLE_PATH.html)
* [CMAKE\_AUTOMOC\_RELAXED\_MODE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_AUTOMOC_RELAXED_MODE.html)
* [CMAKE\_BACKWARDS\_COMPATIBILITY](https://cmake.org/cmake/help/v3.5/variable/CMAKE_BACKWARDS_COMPATIBILITY.html)
* [CMAKE\_BUILD\_TYPE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_BUILD_TYPE.html)
* [CMAKE\_COLOR\_MAKEFILE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_COLOR_MAKEFILE.html)
* [CMAKE\_CONFIGURATION\_TYPES](https://cmake.org/cmake/help/v3.5/variable/CMAKE_CONFIGURATION_TYPES.html)
* [CMAKE\_DEBUG\_TARGET\_PROPERTIES](https://cmake.org/cmake/help/v3.5/variable/CMAKE_DEBUG_TARGET_PROPERTIES.html)
* [CMAKE\_DISABLE\_FIND\_PACKAGE\_<PackageName>](https://cmake.org/cmake/help/v3.5/variable/CMAKE_DISABLE_FIND_PACKAGE_PackageName.html)
* [CMAKE\_ERROR\_DEPRECATED](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ERROR_DEPRECATED.html)
* [CMAKE\_ERROR\_ON\_ABSOLUTE\_INSTALL\_DESTINATION](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ERROR_ON_ABSOLUTE_INSTALL_DESTINATION.html)
* [CMAKE\_EXPORT\_COMPILE\_COMMANDS](https://cmake.org/cmake/help/v3.5/variable/CMAKE_EXPORT_COMPILE_COMMANDS.html)
* [CMAKE\_EXPORT\_NO\_PACKAGE\_REGISTRY](https://cmake.org/cmake/help/v3.5/variable/CMAKE_EXPORT_NO_PACKAGE_REGISTRY.html)
* [CMAKE\_SYSROOT](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SYSROOT.html)
* [CMAKE\_FIND\_APPBUNDLE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_FIND_APPBUNDLE.html)
* [CMAKE\_FIND\_FRAMEWORK](https://cmake.org/cmake/help/v3.5/variable/CMAKE_FIND_FRAMEWORK.html)
* [CMAKE\_FIND\_LIBRARY\_PREFIXES](https://cmake.org/cmake/help/v3.5/variable/CMAKE_FIND_LIBRARY_PREFIXES.html)
* [CMAKE\_FIND\_LIBRARY\_SUFFIXES](https://cmake.org/cmake/help/v3.5/variable/CMAKE_FIND_LIBRARY_SUFFIXES.html)
* [CMAKE\_FIND\_NO\_INSTALL\_PREFIX](https://cmake.org/cmake/help/v3.5/variable/CMAKE_FIND_NO_INSTALL_PREFIX.html)
* [CMAKE\_FIND\_PACKAGE\_NO\_PACKAGE\_REGISTRY](https://cmake.org/cmake/help/v3.5/variable/CMAKE_FIND_PACKAGE_NO_PACKAGE_REGISTRY.html)
* [CMAKE\_FIND\_PACKAGE\_NO\_SYSTEM\_PACKAGE\_REGISTRY](https://cmake.org/cmake/help/v3.5/variable/CMAKE_FIND_PACKAGE_NO_SYSTEM_PACKAGE_REGISTRY.html)
* [CMAKE\_FIND\_PACKAGE\_WARN\_NO\_MODULE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_FIND_PACKAGE_WARN_NO_MODULE.html)
* [CMAKE\_FIND\_ROOT\_PATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_FIND_ROOT_PATH.html)
* [CMAKE\_FIND\_ROOT\_PATH\_MODE\_INCLUDE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_FIND_ROOT_PATH_MODE_INCLUDE.html)
* [CMAKE\_FIND\_ROOT\_PATH\_MODE\_LIBRARY](https://cmake.org/cmake/help/v3.5/variable/CMAKE_FIND_ROOT_PATH_MODE_LIBRARY.html)
* [CMAKE\_FIND\_ROOT\_PATH\_MODE\_PACKAGE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_FIND_ROOT_PATH_MODE_PACKAGE.html)
* [CMAKE\_FIND\_ROOT\_PATH\_MODE\_PROGRAM](https://cmake.org/cmake/help/v3.5/variable/CMAKE_FIND_ROOT_PATH_MODE_PROGRAM.html)
* [CMAKE\_FRAMEWORK\_PATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_FRAMEWORK_PATH.html)
* [CMAKE\_IGNORE\_PATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_IGNORE_PATH.html)
* [CMAKE\_INCLUDE\_PATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_INCLUDE_PATH.html) 环境变量,非cmake变量
* [CMAKE\_INCLUDE\_DIRECTORIES\_BEFORE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_INCLUDE_DIRECTORIES_BEFORE.html)
* [CMAKE\_INCLUDE\_DIRECTORIES\_PROJECT\_BEFORE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_INCLUDE_DIRECTORIES_PROJECT_BEFORE.html)
* [CMAKE\_INSTALL\_DEFAULT\_COMPONENT\_NAME](https://cmake.org/cmake/help/v3.5/variable/CMAKE_INSTALL_DEFAULT_COMPONENT_NAME.html)
* [CMAKE\_INSTALL\_MESSAGE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_INSTALL_MESSAGE.html)
* [CMAKE\_INSTALL\_PREFIX](https://cmake.org/cmake/help/v3.5/variable/CMAKE_INSTALL_PREFIX.html)
* [CMAKE\_LIBRARY\_PATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LIBRARY_PATH.html) 环境变量
* [CMAKE\_MFC\_FLAG](https://cmake.org/cmake/help/v3.5/variable/CMAKE_MFC_FLAG.html)
* [CMAKE\_MODULE\_PATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_MODULE_PATH.html)
* [CMAKE\_NOT\_USING\_CONFIG\_FLAGS](https://cmake.org/cmake/help/v3.5/variable/CMAKE_NOT_USING_CONFIG_FLAGS.html)
* [CMAKE\_POLICY\_DEFAULT\_CMP<NNNN>](https://cmake.org/cmake/help/v3.5/variable/CMAKE_POLICY_DEFAULT_CMPNNNN.html)
* [CMAKE\_POLICY\_WARNING\_CMP<NNNN>](https://cmake.org/cmake/help/v3.5/variable/CMAKE_POLICY_WARNING_CMPNNNN.html)
* [CMAKE\_PREFIX\_PATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_PREFIX_PATH.html)
* [CMAKE\_PROGRAM\_PATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_PROGRAM_PATH.html)
* [CMAKE\_PROJECT\_<PROJECT-NAME>\_INCLUDE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_PROJECT_PROJECT-NAME_INCLUDE.html)
* [CMAKE\_SKIP\_INSTALL\_ALL\_DEPENDENCY](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SKIP_INSTALL_ALL_DEPENDENCY.html)
* [CMAKE\_STAGING\_PREFIX](https://cmake.org/cmake/help/v3.5/variable/CMAKE_STAGING_PREFIX.html)
* [CMAKE\_SYSTEM\_APPBUNDLE\_PATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SYSTEM_APPBUNDLE_PATH.html)
* [CMAKE\_SYSTEM\_FRAMEWORK\_PATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SYSTEM_FRAMEWORK_PATH.html)
* [CMAKE\_SYSTEM\_IGNORE\_PATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SYSTEM_IGNORE_PATH.html)
* [CMAKE\_SYSTEM\_INCLUDE\_PATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SYSTEM_INCLUDE_PATH.html)
* [CMAKE\_SYSTEM\_LIBRARY\_PATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SYSTEM_LIBRARY_PATH.html)
* [CMAKE\_SYSTEM\_PREFIX\_PATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SYSTEM_PREFIX_PATH.html)
* [CMAKE\_SYSTEM\_PROGRAM\_PATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SYSTEM_PROGRAM_PATH.html)
* [CMAKE\_USER\_MAKE\_RULES\_OVERRIDE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_USER_MAKE_RULES_OVERRIDE.html)
* [CMAKE\_WARN\_DEPRECATED](https://cmake.org/cmake/help/v3.5/variable/CMAKE_WARN_DEPRECATED.html)
* [CMAKE\_WARN\_ON\_ABSOLUTE\_INSTALL\_DESTINATION](https://cmake.org/cmake/help/v3.5/variable/CMAKE_WARN_ON_ABSOLUTE_INSTALL_DESTINATION.html)

###### Variables that Describe the System

* [APPLE](https://cmake.org/cmake/help/v3.5/variable/APPLE.html)
* [BORLAND](https://cmake.org/cmake/help/v3.5/variable/BORLAND.html)
* [CMAKE\_CL\_64](https://cmake.org/cmake/help/v3.5/variable/CMAKE_CL_64.html)
* [CMAKE\_COMPILER\_2005](https://cmake.org/cmake/help/v3.5/variable/CMAKE_COMPILER_2005.html)
* [CMAKE\_HOST\_APPLE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_HOST_APPLE.html)
* [CMAKE\_HOST\_SYSTEM\_NAME](https://cmake.org/cmake/help/v3.5/variable/CMAKE_HOST_SYSTEM_NAME.html)
* [CMAKE\_HOST\_SYSTEM\_PROCESSOR](https://cmake.org/cmake/help/v3.5/variable/CMAKE_HOST_SYSTEM_PROCESSOR.html)
* [CMAKE\_HOST\_SYSTEM](https://cmake.org/cmake/help/v3.5/variable/CMAKE_HOST_SYSTEM.html)
* [CMAKE\_HOST\_SYSTEM\_VERSION](https://cmake.org/cmake/help/v3.5/variable/CMAKE_HOST_SYSTEM_VERSION.html)
* [CMAKE\_HOST\_UNIX](https://cmake.org/cmake/help/v3.5/variable/CMAKE_HOST_UNIX.html)
* [CMAKE\_HOST\_WIN32](https://cmake.org/cmake/help/v3.5/variable/CMAKE_HOST_WIN32.html)
* [CMAKE\_LIBRARY\_ARCHITECTURE\_REGEX](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LIBRARY_ARCHITECTURE_REGEX.html)
* [CMAKE\_LIBRARY\_ARCHITECTURE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LIBRARY_ARCHITECTURE.html)
* [CMAKE\_OBJECT\_PATH\_MAX](https://cmake.org/cmake/help/v3.5/variable/CMAKE_OBJECT_PATH_MAX.html)
* [CMAKE\_SYSTEM\_NAME](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SYSTEM_NAME.html)
* [CMAKE\_SYSTEM\_PROCESSOR](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SYSTEM_PROCESSOR.html)
* [CMAKE\_SYSTEM](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SYSTEM.html)
* [CMAKE\_SYSTEM\_VERSION](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SYSTEM_VERSION.html)
* [CYGWIN](https://cmake.org/cmake/help/v3.5/variable/CYGWIN.html)
* [ENV](https://cmake.org/cmake/help/v3.5/variable/ENV.html)
* [GHS-MULTI](https://cmake.org/cmake/help/v3.5/variable/GHS-MULTI.html)
* [MINGW](https://cmake.org/cmake/help/v3.5/variable/MINGW.html)
* [MSVC10](https://cmake.org/cmake/help/v3.5/variable/MSVC10.html)
* [MSVC11](https://cmake.org/cmake/help/v3.5/variable/MSVC11.html)
* [MSVC12](https://cmake.org/cmake/help/v3.5/variable/MSVC12.html)
* [MSVC14](https://cmake.org/cmake/help/v3.5/variable/MSVC14.html)
* [MSVC60](https://cmake.org/cmake/help/v3.5/variable/MSVC60.html)
* [MSVC70](https://cmake.org/cmake/help/v3.5/variable/MSVC70.html)
* [MSVC71](https://cmake.org/cmake/help/v3.5/variable/MSVC71.html)
* [MSVC80](https://cmake.org/cmake/help/v3.5/variable/MSVC80.html)
* [MSVC90](https://cmake.org/cmake/help/v3.5/variable/MSVC90.html)
* [MSVC\_IDE](https://cmake.org/cmake/help/v3.5/variable/MSVC_IDE.html)
* [MSVC](https://cmake.org/cmake/help/v3.5/variable/MSVC.html)
* [MSVC\_VERSION](https://cmake.org/cmake/help/v3.5/variable/MSVC_VERSION.html)
* [UNIX](https://cmake.org/cmake/help/v3.5/variable/UNIX.html)
* [WIN32](https://cmake.org/cmake/help/v3.5/variable/WIN32.html)
* [WINCE](https://cmake.org/cmake/help/v3.5/variable/WINCE.html)
* [WINDOWS\_PHONE](https://cmake.org/cmake/help/v3.5/variable/WINDOWS_PHONE.html)
* [WINDOWS\_STORE](https://cmake.org/cmake/help/v3.5/variable/WINDOWS_STORE.html)
* [XCODE\_VERSION](https://cmake.org/cmake/help/v3.5/variable/XCODE_VERSION.html)

###### Variables that Control the Build

* [CMAKE\_ANDROID\_ANT\_ADDITIONAL\_OPTIONS](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ANDROID_ANT_ADDITIONAL_OPTIONS.html)
* [CMAKE\_ANDROID\_API](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ANDROID_API.html)
* [CMAKE\_ANDROID\_API\_MIN](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ANDROID_API_MIN.html)
* [CMAKE\_ANDROID\_ARCH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ANDROID_ARCH.html)
* [CMAKE\_ANDROID\_ASSETS\_DIRECTORIES](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ANDROID_ASSETS_DIRECTORIES.html)
* [CMAKE\_ANDROID\_GUI](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ANDROID_GUI.html)
* [CMAKE\_ANDROID\_JAR\_DEPENDENCIES](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ANDROID_JAR_DEPENDENCIES.html)
* [CMAKE\_ANDROID\_JAR\_DIRECTORIES](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ANDROID_JAR_DIRECTORIES.html)
* [CMAKE\_ANDROID\_JAVA\_SOURCE\_DIR](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ANDROID_JAVA_SOURCE_DIR.html)
* [CMAKE\_ANDROID\_NATIVE\_LIB\_DEPENDENCIES](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ANDROID_NATIVE_LIB_DEPENDENCIES.html)
* [CMAKE\_ANDROID\_NATIVE\_LIB\_DIRECTORIES](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ANDROID_NATIVE_LIB_DIRECTORIES.html)
* [CMAKE\_ANDROID\_PROCESS\_MAX](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ANDROID_PROCESS_MAX.html)
* [CMAKE\_ANDROID\_PROGUARD](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ANDROID_PROGUARD.html)
* [CMAKE\_ANDROID\_PROGUARD\_CONFIG\_PATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ANDROID_PROGUARD_CONFIG_PATH.html)
* [CMAKE\_ANDROID\_SECURE\_PROPS\_PATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ANDROID_SECURE_PROPS_PATH.html)
* [CMAKE\_ANDROID\_SKIP\_ANT\_STEP](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ANDROID_SKIP_ANT_STEP.html)
* [CMAKE\_ANDROID\_STL\_TYPE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ANDROID_STL_TYPE.html)
* [CMAKE\_ARCHIVE\_OUTPUT\_DIRECTORY](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ARCHIVE_OUTPUT_DIRECTORY.html)
* [CMAKE\_ARCHIVE\_OUTPUT\_DIRECTORY\_<CONFIG>](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ARCHIVE_OUTPUT_DIRECTORY_CONFIG.html)
* [CMAKE\_AUTOMOC\_MOC\_OPTIONS](https://cmake.org/cmake/help/v3.5/variable/CMAKE_AUTOMOC_MOC_OPTIONS.html)
* [CMAKE\_AUTOMOC](https://cmake.org/cmake/help/v3.5/variable/CMAKE_AUTOMOC.html)
* [CMAKE\_AUTORCC](https://cmake.org/cmake/help/v3.5/variable/CMAKE_AUTORCC.html)
* [CMAKE\_AUTORCC\_OPTIONS](https://cmake.org/cmake/help/v3.5/variable/CMAKE_AUTORCC_OPTIONS.html)
* [CMAKE\_AUTOUIC](https://cmake.org/cmake/help/v3.5/variable/CMAKE_AUTOUIC.html)
* [CMAKE\_AUTOUIC\_OPTIONS](https://cmake.org/cmake/help/v3.5/variable/CMAKE_AUTOUIC_OPTIONS.html)
* [CMAKE\_BUILD\_WITH\_INSTALL\_RPATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_BUILD_WITH_INSTALL_RPATH.html)
* [CMAKE\_COMPILE\_PDB\_OUTPUT\_DIRECTORY](https://cmake.org/cmake/help/v3.5/variable/CMAKE_COMPILE_PDB_OUTPUT_DIRECTORY.html)
* [CMAKE\_COMPILE\_PDB\_OUTPUT\_DIRECTORY\_<CONFIG>](https://cmake.org/cmake/help/v3.5/variable/CMAKE_COMPILE_PDB_OUTPUT_DIRECTORY_CONFIG.html)
* [CMAKE\_<CONFIG>\_POSTFIX](https://cmake.org/cmake/help/v3.5/variable/CMAKE_CONFIG_POSTFIX.html)
* [CMAKE\_DEBUG\_POSTFIX](https://cmake.org/cmake/help/v3.5/variable/CMAKE_DEBUG_POSTFIX.html)
* [CMAKE\_ENABLE\_EXPORTS](https://cmake.org/cmake/help/v3.5/variable/CMAKE_ENABLE_EXPORTS.html)
* [CMAKE\_EXE\_LINKER\_FLAGS\_<CONFIG>](https://cmake.org/cmake/help/v3.5/variable/CMAKE_EXE_LINKER_FLAGS_CONFIG.html)
* [CMAKE\_EXE\_LINKER\_FLAGS](https://cmake.org/cmake/help/v3.5/variable/CMAKE_EXE_LINKER_FLAGS.html)
* [CMAKE\_Fortran\_FORMAT](https://cmake.org/cmake/help/v3.5/variable/CMAKE_Fortran_FORMAT.html)
* [CMAKE\_Fortran\_MODULE\_DIRECTORY](https://cmake.org/cmake/help/v3.5/variable/CMAKE_Fortran_MODULE_DIRECTORY.html)
* [CMAKE\_GNUtoMS](https://cmake.org/cmake/help/v3.5/variable/CMAKE_GNUtoMS.html)
* [CMAKE\_INCLUDE\_CURRENT\_DIR\_IN\_INTERFACE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_INCLUDE_CURRENT_DIR_IN_INTERFACE.html)
* [CMAKE\_INCLUDE\_CURRENT\_DIR](https://cmake.org/cmake/help/v3.5/variable/CMAKE_INCLUDE_CURRENT_DIR.html)
* [CMAKE\_INSTALL\_NAME\_DIR](https://cmake.org/cmake/help/v3.5/variable/CMAKE_INSTALL_NAME_DIR.html)
* [CMAKE\_INSTALL\_RPATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_INSTALL_RPATH.html)
* [CMAKE\_INSTALL\_RPATH\_USE\_LINK\_PATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_INSTALL_RPATH_USE_LINK_PATH.html)
* [CMAKE\_IOS\_INSTALL\_COMBINED](https://cmake.org/cmake/help/v3.5/variable/CMAKE_IOS_INSTALL_COMBINED.html)
* [CMAKE\_<LANG>\_COMPILER\_LAUNCHER](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_COMPILER_LAUNCHER.html)
* [CMAKE\_<LANG>\_INCLUDE\_WHAT\_YOU\_USE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_INCLUDE_WHAT_YOU_USE.html)
* [CMAKE\_<LANG>\_VISIBILITY\_PRESET](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_VISIBILITY_PRESET.html)
* [CMAKE\_LIBRARY\_OUTPUT\_DIRECTORY](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LIBRARY_OUTPUT_DIRECTORY.html)
* [CMAKE\_LIBRARY\_OUTPUT\_DIRECTORY\_<CONFIG>](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LIBRARY_OUTPUT_DIRECTORY_CONFIG.html)
* [CMAKE\_LIBRARY\_PATH\_FLAG](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LIBRARY_PATH_FLAG.html)
* [CMAKE\_LINK\_DEF\_FILE\_FLAG](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LINK_DEF_FILE_FLAG.html)
* [CMAKE\_LINK\_DEPENDS\_NO\_SHARED](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LINK_DEPENDS_NO_SHARED.html)
* [CMAKE\_LINK\_INTERFACE\_LIBRARIES](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LINK_INTERFACE_LIBRARIES.html)
* [CMAKE\_LINK\_LIBRARY\_FILE\_FLAG](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LINK_LIBRARY_FILE_FLAG.html)
* [CMAKE\_LINK\_LIBRARY\_FLAG](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LINK_LIBRARY_FLAG.html)
* [CMAKE\_MACOSX\_BUNDLE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_MACOSX_BUNDLE.html)
* [CMAKE\_MACOSX\_RPATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_MACOSX_RPATH.html)
* [CMAKE\_MAP\_IMPORTED\_CONFIG\_<CONFIG>](https://cmake.org/cmake/help/v3.5/variable/CMAKE_MAP_IMPORTED_CONFIG_CONFIG.html)
* [CMAKE\_MODULE\_LINKER\_FLAGS\_<CONFIG>](https://cmake.org/cmake/help/v3.5/variable/CMAKE_MODULE_LINKER_FLAGS_CONFIG.html)
* [CMAKE\_MODULE\_LINKER\_FLAGS](https://cmake.org/cmake/help/v3.5/variable/CMAKE_MODULE_LINKER_FLAGS.html)
* [CMAKE\_NO\_BUILTIN\_CHRPATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_NO_BUILTIN_CHRPATH.html)
* [CMAKE\_NO\_SYSTEM\_FROM\_IMPORTED](https://cmake.org/cmake/help/v3.5/variable/CMAKE_NO_SYSTEM_FROM_IMPORTED.html)
* [CMAKE\_OSX\_ARCHITECTURES](https://cmake.org/cmake/help/v3.5/variable/CMAKE_OSX_ARCHITECTURES.html)
* [CMAKE\_OSX\_DEPLOYMENT\_TARGET](https://cmake.org/cmake/help/v3.5/variable/CMAKE_OSX_DEPLOYMENT_TARGET.html)
* [CMAKE\_OSX\_SYSROOT](https://cmake.org/cmake/help/v3.5/variable/CMAKE_OSX_SYSROOT.html)
* [CMAKE\_PDB\_OUTPUT\_DIRECTORY](https://cmake.org/cmake/help/v3.5/variable/CMAKE_PDB_OUTPUT_DIRECTORY.html)
* [CMAKE\_PDB\_OUTPUT\_DIRECTORY\_<CONFIG>](https://cmake.org/cmake/help/v3.5/variable/CMAKE_PDB_OUTPUT_DIRECTORY_CONFIG.html)
* [CMAKE\_POSITION\_INDEPENDENT\_CODE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_POSITION_INDEPENDENT_CODE.html)
* [CMAKE\_RUNTIME\_OUTPUT\_DIRECTORY](https://cmake.org/cmake/help/v3.5/variable/CMAKE_RUNTIME_OUTPUT_DIRECTORY.html)
* [CMAKE\_RUNTIME\_OUTPUT\_DIRECTORY\_<CONFIG>](https://cmake.org/cmake/help/v3.5/variable/CMAKE_RUNTIME_OUTPUT_DIRECTORY_CONFIG.html)
* [CMAKE\_SHARED\_LINKER\_FLAGS\_<CONFIG>](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SHARED_LINKER_FLAGS_CONFIG.html)
* [CMAKE\_SHARED\_LINKER\_FLAGS](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SHARED_LINKER_FLAGS.html)
* [CMAKE\_SKIP\_BUILD\_RPATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SKIP_BUILD_RPATH.html)
* [CMAKE\_SKIP\_INSTALL\_RPATH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_SKIP_INSTALL_RPATH.html)
* [CMAKE\_STATIC\_LINKER\_FLAGS\_<CONFIG>](https://cmake.org/cmake/help/v3.5/variable/CMAKE_STATIC_LINKER_FLAGS_CONFIG.html)
* [CMAKE\_STATIC\_LINKER\_FLAGS](https://cmake.org/cmake/help/v3.5/variable/CMAKE_STATIC_LINKER_FLAGS.html)
* [CMAKE\_TRY\_COMPILE\_CONFIGURATION](https://cmake.org/cmake/help/v3.5/variable/CMAKE_TRY_COMPILE_CONFIGURATION.html)
* [CMAKE\_USE\_RELATIVE\_PATHS](https://cmake.org/cmake/help/v3.5/variable/CMAKE_USE_RELATIVE_PATHS.html)
* [CMAKE\_VISIBILITY\_INLINES\_HIDDEN](https://cmake.org/cmake/help/v3.5/variable/CMAKE_VISIBILITY_INLINES_HIDDEN.html)
* [CMAKE\_VS\_INCLUDE\_INSTALL\_TO\_DEFAULT\_BUILD](https://cmake.org/cmake/help/v3.5/variable/CMAKE_VS_INCLUDE_INSTALL_TO_DEFAULT_BUILD.html)
* [CMAKE\_WINDOWS\_EXPORT\_ALL\_SYMBOLS](https://cmake.org/cmake/help/v3.5/variable/CMAKE_WINDOWS_EXPORT_ALL_SYMBOLS.html)
* [CMAKE\_WIN32\_EXECUTABLE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_WIN32_EXECUTABLE.html)
* [CMAKE\_XCODE\_ATTRIBUTE\_<an-attribute>](https://cmake.org/cmake/help/v3.5/variable/CMAKE_XCODE_ATTRIBUTE_an-attribute.html)
* [EXECUTABLE\_OUTPUT\_PATH](https://cmake.org/cmake/help/v3.5/variable/EXECUTABLE_OUTPUT_PATH.html)
* [LIBRARY\_OUTPUT\_PATH](https://cmake.org/cmake/help/v3.5/variable/LIBRARY_OUTPUT_PATH.html)

###### Variables for Languages

* [CMAKE\_COMPILER\_IS\_GNU<LANG>](https://cmake.org/cmake/help/v3.5/variable/CMAKE_COMPILER_IS_GNULANG.html)
* [CMAKE\_C\_COMPILE\_FEATURES](https://cmake.org/cmake/help/v3.5/variable/CMAKE_C_COMPILE_FEATURES.html)
* [CMAKE\_C\_EXTENSIONS](https://cmake.org/cmake/help/v3.5/variable/CMAKE_C_EXTENSIONS.html)
* [CMAKE\_C\_STANDARD](https://cmake.org/cmake/help/v3.5/variable/CMAKE_C_STANDARD.html)
* [CMAKE\_C\_STANDARD\_REQUIRED](https://cmake.org/cmake/help/v3.5/variable/CMAKE_C_STANDARD_REQUIRED.html)
* [CMAKE\_CXX\_COMPILE\_FEATURES](https://cmake.org/cmake/help/v3.5/variable/CMAKE_CXX_COMPILE_FEATURES.html)
* [CMAKE\_CXX\_EXTENSIONS](https://cmake.org/cmake/help/v3.5/variable/CMAKE_CXX_EXTENSIONS.html)
* [CMAKE\_CXX\_STANDARD](https://cmake.org/cmake/help/v3.5/variable/CMAKE_CXX_STANDARD.html)
* [CMAKE\_CXX\_STANDARD\_REQUIRED](https://cmake.org/cmake/help/v3.5/variable/CMAKE_CXX_STANDARD_REQUIRED.html)
* [CMAKE\_Fortran\_MODDIR\_DEFAULT](https://cmake.org/cmake/help/v3.5/variable/CMAKE_Fortran_MODDIR_DEFAULT.html)
* [CMAKE\_Fortran\_MODDIR\_FLAG](https://cmake.org/cmake/help/v3.5/variable/CMAKE_Fortran_MODDIR_FLAG.html)
* [CMAKE\_Fortran\_MODOUT\_FLAG](https://cmake.org/cmake/help/v3.5/variable/CMAKE_Fortran_MODOUT_FLAG.html)
* [CMAKE\_INTERNAL\_PLATFORM\_ABI](https://cmake.org/cmake/help/v3.5/variable/CMAKE_INTERNAL_PLATFORM_ABI.html)
* [CMAKE\_<LANG>\_ARCHIVE\_APPEND](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_ARCHIVE_APPEND.html)
* [CMAKE\_<LANG>\_ARCHIVE\_CREATE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_ARCHIVE_CREATE.html)
* [CMAKE\_<LANG>\_ARCHIVE\_FINISH](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_ARCHIVE_FINISH.html)
* [CMAKE\_<LANG>\_COMPILE\_OBJECT](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_COMPILE_OBJECT.html)
* [CMAKE\_<LANG>\_COMPILER\_ABI](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_COMPILER_ABI.html)
* [CMAKE\_<LANG>\_COMPILER\_ID](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_COMPILER_ID.html)
* [CMAKE\_<LANG>\_COMPILER\_LOADED](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_COMPILER_LOADED.html)
* [CMAKE\_<LANG>\_COMPILER](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_COMPILER.html)
* [CMAKE\_<LANG>\_COMPILER\_EXTERNAL\_TOOLCHAIN](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_COMPILER_EXTERNAL_TOOLCHAIN.html)
* [CMAKE\_<LANG>\_COMPILER\_TARGET](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_COMPILER_TARGET.html)
* [CMAKE\_<LANG>\_COMPILER\_VERSION](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_COMPILER_VERSION.html)
* [CMAKE\_<LANG>\_CREATE\_SHARED\_LIBRARY](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_CREATE_SHARED_LIBRARY.html)
* [CMAKE\_<LANG>\_CREATE\_SHARED\_MODULE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_CREATE_SHARED_MODULE.html)
* [CMAKE\_<LANG>\_CREATE\_STATIC\_LIBRARY](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_CREATE_STATIC_LIBRARY.html)
* [CMAKE\_<LANG>\_FLAGS\_DEBUG](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_FLAGS_DEBUG.html)
* [CMAKE\_<LANG>\_FLAGS\_MINSIZEREL](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_FLAGS_MINSIZEREL.html)
* [CMAKE\_<LANG>\_FLAGS\_RELEASE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_FLAGS_RELEASE.html)
* [CMAKE\_<LANG>\_FLAGS\_RELWITHDEBINFO](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_FLAGS_RELWITHDEBINFO.html)
* [CMAKE\_<LANG>\_FLAGS](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_FLAGS.html)
* [CMAKE\_<LANG>\_GHS\_KERNEL\_FLAGS\_DEBUG](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_GHS_KERNEL_FLAGS_DEBUG.html)
* [CMAKE\_<LANG>\_GHS\_KERNEL\_FLAGS\_MINSIZEREL](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_GHS_KERNEL_FLAGS_MINSIZEREL.html)
* [CMAKE\_<LANG>\_GHS\_KERNEL\_FLAGS\_RELEASE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_GHS_KERNEL_FLAGS_RELEASE.html)
* [CMAKE\_<LANG>\_GHS\_KERNEL\_FLAGS\_RELWITHDEBINFO](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_GHS_KERNEL_FLAGS_RELWITHDEBINFO.html)
* [CMAKE\_<LANG>\_IGNORE\_EXTENSIONS](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_IGNORE_EXTENSIONS.html)
* [CMAKE\_<LANG>\_IMPLICIT\_INCLUDE\_DIRECTORIES](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_IMPLICIT_INCLUDE_DIRECTORIES.html)
* [CMAKE\_<LANG>\_IMPLICIT\_LINK\_DIRECTORIES](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_IMPLICIT_LINK_DIRECTORIES.html)
* [CMAKE\_<LANG>\_IMPLICIT\_LINK\_FRAMEWORK\_DIRECTORIES](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_IMPLICIT_LINK_FRAMEWORK_DIRECTORIES.html)
* [CMAKE\_<LANG>\_IMPLICIT\_LINK\_LIBRARIES](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_IMPLICIT_LINK_LIBRARIES.html)
* [CMAKE\_<LANG>\_LIBRARY\_ARCHITECTURE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_LIBRARY_ARCHITECTURE.html)
* [CMAKE\_<LANG>\_LINKER\_PREFERENCE\_PROPAGATES](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_LINKER_PREFERENCE_PROPAGATES.html)
* [CMAKE\_<LANG>\_LINKER\_PREFERENCE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_LINKER_PREFERENCE.html)
* [CMAKE\_<LANG>\_LINK\_EXECUTABLE](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_LINK_EXECUTABLE.html)
* [CMAKE\_<LANG>\_OUTPUT\_EXTENSION](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_OUTPUT_EXTENSION.html)
* [CMAKE\_<LANG>\_PLATFORM\_ID](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_PLATFORM_ID.html)
* [CMAKE\_<LANG>\_SIMULATE\_ID](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_SIMULATE_ID.html)
* [CMAKE\_<LANG>\_SIMULATE\_VERSION](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_SIMULATE_VERSION.html)
* [CMAKE\_<LANG>\_SIZEOF\_DATA\_PTR](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_SIZEOF_DATA_PTR.html)
* [CMAKE\_<LANG>\_SOURCE\_FILE\_EXTENSIONS](https://cmake.org/cmake/help/v3.5/variable/CMAKE_LANG_SOURCE_FILE_EXTENSIONS.html)
* [CMAKE\_USER\_MAKE\_RULES\_OVERRIDE\_<LANG>](https://cmake.org/cmake/help/v3.5/variable/CMAKE_USER_MAKE_RULES_OVERRIDE_LANG.html)

###### Variables for CTest

* [CTEST\_BINARY\_DIRECTORY](https://cmake.org/cmake/help/v3.5/variable/CTEST_BINARY_DIRECTORY.html)
* [CTEST\_BUILD\_COMMAND](https://cmake.org/cmake/help/v3.5/variable/CTEST_BUILD_COMMAND.html)
* [CTEST\_BUILD\_NAME](https://cmake.org/cmake/help/v3.5/variable/CTEST_BUILD_NAME.html)
* [CTEST\_BZR\_COMMAND](https://cmake.org/cmake/help/v3.5/variable/CTEST_BZR_COMMAND.html)
* [CTEST\_BZR\_UPDATE\_OPTIONS](https://cmake.org/cmake/help/v3.5/variable/CTEST_BZR_UPDATE_OPTIONS.html)
* [CTEST\_CHANGE\_ID](https://cmake.org/cmake/help/v3.5/variable/CTEST_CHANGE_ID.html)
* [CTEST\_CHECKOUT\_COMMAND](https://cmake.org/cmake/help/v3.5/variable/CTEST_CHECKOUT_COMMAND.html)
* [CTEST\_CONFIGURATION\_TYPE](https://cmake.org/cmake/help/v3.5/variable/CTEST_CONFIGURATION_TYPE.html)
* [CTEST\_CONFIGURE\_COMMAND](https://cmake.org/cmake/help/v3.5/variable/CTEST_CONFIGURE_COMMAND.html)
* [CTEST\_COVERAGE\_COMMAND](https://cmake.org/cmake/help/v3.5/variable/CTEST_COVERAGE_COMMAND.html)
* [CTEST\_COVERAGE\_EXTRA\_FLAGS](https://cmake.org/cmake/help/v3.5/variable/CTEST_COVERAGE_EXTRA_FLAGS.html)
* [CTEST\_CURL\_OPTIONS](https://cmake.org/cmake/help/v3.5/variable/CTEST_CURL_OPTIONS.html)
* [CTEST\_CUSTOM\_COVERAGE\_EXCLUDE](https://cmake.org/cmake/help/v3.5/variable/CTEST_CUSTOM_COVERAGE_EXCLUDE.html)
* [CTEST\_CUSTOM\_ERROR\_EXCEPTION](https://cmake.org/cmake/help/v3.5/variable/CTEST_CUSTOM_ERROR_EXCEPTION.html)
* [CTEST\_CUSTOM\_ERROR\_MATCH](https://cmake.org/cmake/help/v3.5/variable/CTEST_CUSTOM_ERROR_MATCH.html)
* [CTEST\_CUSTOM\_ERROR\_POST\_CONTEXT](https://cmake.org/cmake/help/v3.5/variable/CTEST_CUSTOM_ERROR_POST_CONTEXT.html)
* [CTEST\_CUSTOM\_ERROR\_PRE\_CONTEXT](https://cmake.org/cmake/help/v3.5/variable/CTEST_CUSTOM_ERROR_PRE_CONTEXT.html)
* [CTEST\_CUSTOM\_MAXIMUM\_FAILED\_TEST\_OUTPUT\_SIZE](https://cmake.org/cmake/help/v3.5/variable/CTEST_CUSTOM_MAXIMUM_FAILED_TEST_OUTPUT_SIZE.html)
* [CTEST\_CUSTOM\_MAXIMUM\_NUMBER\_OF\_ERRORS](https://cmake.org/cmake/help/v3.5/variable/CTEST_CUSTOM_MAXIMUM_NUMBER_OF_ERRORS.html)
* [CTEST\_CUSTOM\_MAXIMUM\_NUMBER\_OF\_WARNINGS](https://cmake.org/cmake/help/v3.5/variable/CTEST_CUSTOM_MAXIMUM_NUMBER_OF_WARNINGS.html)
* [CTEST\_CUSTOM\_MAXIMUM\_PASSED\_TEST\_OUTPUT\_SIZE](https://cmake.org/cmake/help/v3.5/variable/CTEST_CUSTOM_MAXIMUM_PASSED_TEST_OUTPUT_SIZE.html)
* [CTEST\_CUSTOM\_MEMCHECK\_IGNORE](https://cmake.org/cmake/help/v3.5/variable/CTEST_CUSTOM_MEMCHECK_IGNORE.html)
* [CTEST\_CUSTOM\_POST\_MEMCHECK](https://cmake.org/cmake/help/v3.5/variable/CTEST_CUSTOM_POST_MEMCHECK.html)
* [CTEST\_CUSTOM\_POST\_TEST](https://cmake.org/cmake/help/v3.5/variable/CTEST_CUSTOM_POST_TEST.html)
* [CTEST\_CUSTOM\_PRE\_MEMCHECK](https://cmake.org/cmake/help/v3.5/variable/CTEST_CUSTOM_PRE_MEMCHECK.html)
* [CTEST\_CUSTOM\_PRE\_TEST](https://cmake.org/cmake/help/v3.5/variable/CTEST_CUSTOM_PRE_TEST.html)
* [CTEST\_CUSTOM\_TEST\_IGNORE](https://cmake.org/cmake/help/v3.5/variable/CTEST_CUSTOM_TEST_IGNORE.html)
* [CTEST\_CUSTOM\_WARNING\_EXCEPTION](https://cmake.org/cmake/help/v3.5/variable/CTEST_CUSTOM_WARNING_EXCEPTION.html)
* [CTEST\_CUSTOM\_WARNING\_MATCH](https://cmake.org/cmake/help/v3.5/variable/CTEST_CUSTOM_WARNING_MATCH.html)
* [CTEST\_CVS\_CHECKOUT](https://cmake.org/cmake/help/v3.5/variable/CTEST_CVS_CHECKOUT.html)
* [CTEST\_CVS\_COMMAND](https://cmake.org/cmake/help/v3.5/variable/CTEST_CVS_COMMAND.html)
* [CTEST\_CVS\_UPDATE\_OPTIONS](https://cmake.org/cmake/help/v3.5/variable/CTEST_CVS_UPDATE_OPTIONS.html)
* [CTEST\_DROP\_LOCATION](https://cmake.org/cmake/help/v3.5/variable/CTEST_DROP_LOCATION.html)
* [CTEST\_DROP\_METHOD](https://cmake.org/cmake/help/v3.5/variable/CTEST_DROP_METHOD.html)
* [CTEST\_DROP\_SITE](https://cmake.org/cmake/help/v3.5/variable/CTEST_DROP_SITE.html)
* [CTEST\_DROP\_SITE\_CDASH](https://cmake.org/cmake/help/v3.5/variable/CTEST_DROP_SITE_CDASH.html)
* [CTEST\_DROP\_SITE\_PASSWORD](https://cmake.org/cmake/help/v3.5/variable/CTEST_DROP_SITE_PASSWORD.html)
* [CTEST\_DROP\_SITE\_USER](https://cmake.org/cmake/help/v3.5/variable/CTEST_DROP_SITE_USER.html)
* [CTEST\_EXTRA\_COVERAGE\_GLOB](https://cmake.org/cmake/help/v3.5/variable/CTEST_EXTRA_COVERAGE_GLOB.html)
* [CTEST\_GIT\_COMMAND](https://cmake.org/cmake/help/v3.5/variable/CTEST_GIT_COMMAND.html)
* [CTEST\_GIT\_UPDATE\_CUSTOM](https://cmake.org/cmake/help/v3.5/variable/CTEST_GIT_UPDATE_CUSTOM.html)
* [CTEST\_GIT\_UPDATE\_OPTIONS](https://cmake.org/cmake/help/v3.5/variable/CTEST_GIT_UPDATE_OPTIONS.html)
* [CTEST\_HG\_COMMAND](https://cmake.org/cmake/help/v3.5/variable/CTEST_HG_COMMAND.html)
* [CTEST\_HG\_UPDATE\_OPTIONS](https://cmake.org/cmake/help/v3.5/variable/CTEST_HG_UPDATE_OPTIONS.html)
* [CTEST\_MEMORYCHECK\_COMMAND](https://cmake.org/cmake/help/v3.5/variable/CTEST_MEMORYCHECK_COMMAND.html)
* [CTEST\_MEMORYCHECK\_COMMAND\_OPTIONS](https://cmake.org/cmake/help/v3.5/variable/CTEST_MEMORYCHECK_COMMAND_OPTIONS.html)
* [CTEST\_MEMORYCHECK\_SANITIZER\_OPTIONS](https://cmake.org/cmake/help/v3.5/variable/CTEST_MEMORYCHECK_SANITIZER_OPTIONS.html)
* [CTEST\_MEMORYCHECK\_SUPPRESSIONS\_FILE](https://cmake.org/cmake/help/v3.5/variable/CTEST_MEMORYCHECK_SUPPRESSIONS_FILE.html)
* [CTEST\_MEMORYCHECK\_TYPE](https://cmake.org/cmake/help/v3.5/variable/CTEST_MEMORYCHECK_TYPE.html)
* [CTEST\_NIGHTLY\_START\_TIME](https://cmake.org/cmake/help/v3.5/variable/CTEST_NIGHTLY_START_TIME.html)
* [CTEST\_P4\_CLIENT](https://cmake.org/cmake/help/v3.5/variable/CTEST_P4_CLIENT.html)
* [CTEST\_P4\_COMMAND](https://cmake.org/cmake/help/v3.5/variable/CTEST_P4_COMMAND.html)
* [CTEST\_P4\_OPTIONS](https://cmake.org/cmake/help/v3.5/variable/CTEST_P4_OPTIONS.html)
* [CTEST\_P4\_UPDATE\_OPTIONS](https://cmake.org/cmake/help/v3.5/variable/CTEST_P4_UPDATE_OPTIONS.html)
* [CTEST\_SCP\_COMMAND](https://cmake.org/cmake/help/v3.5/variable/CTEST_SCP_COMMAND.html)
* [CTEST\_SITE](https://cmake.org/cmake/help/v3.5/variable/CTEST_SITE.html)
* [CTEST\_SOURCE\_DIRECTORY](https://cmake.org/cmake/help/v3.5/variable/CTEST_SOURCE_DIRECTORY.html)
* [CTEST\_SVN\_COMMAND](https://cmake.org/cmake/help/v3.5/variable/CTEST_SVN_COMMAND.html)
* [CTEST\_SVN\_OPTIONS](https://cmake.org/cmake/help/v3.5/variable/CTEST_SVN_OPTIONS.html)
* [CTEST\_SVN\_UPDATE\_OPTIONS](https://cmake.org/cmake/help/v3.5/variable/CTEST_SVN_UPDATE_OPTIONS.html)
* [CTEST\_TEST\_LOAD](https://cmake.org/cmake/help/v3.5/variable/CTEST_TEST_LOAD.html)
* [CTEST\_TEST\_TIMEOUT](https://cmake.org/cmake/help/v3.5/variable/CTEST_TEST_TIMEOUT.html)
* [CTEST\_TRIGGER\_SITE](https://cmake.org/cmake/help/v3.5/variable/CTEST_TRIGGER_SITE.html)
* [CTEST\_UPDATE\_COMMAND](https://cmake.org/cmake/help/v3.5/variable/CTEST_UPDATE_COMMAND.html)
* [CTEST\_UPDATE\_OPTIONS](https://cmake.org/cmake/help/v3.5/variable/CTEST_UPDATE_OPTIONS.html)
* [CTEST\_UPDATE\_VERSION\_ONLY](https://cmake.org/cmake/help/v3.5/variable/CTEST_UPDATE_VERSION_ONLY.html)
* [CTEST\_USE\_LAUNCHERS](https://cmake.org/cmake/help/v3.5/variable/CTEST_USE_LAUNCHERS.html)

###### Variables for CPack

* [CPACK\_ABSOLUTE\_DESTINATION\_FILES](https://cmake.org/cmake/help/v3.5/variable/CPACK_ABSOLUTE_DESTINATION_FILES.html)
* [CPACK\_COMPONENT\_INCLUDE\_TOPLEVEL\_DIRECTORY](https://cmake.org/cmake/help/v3.5/variable/CPACK_COMPONENT_INCLUDE_TOPLEVEL_DIRECTORY.html)
* [CPACK\_ERROR\_ON\_ABSOLUTE\_INSTALL\_DESTINATION](https://cmake.org/cmake/help/v3.5/variable/CPACK_ERROR_ON_ABSOLUTE_INSTALL_DESTINATION.html)
* [CPACK\_INCLUDE\_TOPLEVEL\_DIRECTORY](https://cmake.org/cmake/help/v3.5/variable/CPACK_INCLUDE_TOPLEVEL_DIRECTORY.html)
* [CPACK\_INSTALL\_SCRIPT](https://cmake.org/cmake/help/v3.5/variable/CPACK_INSTALL_SCRIPT.html)
* [CPACK\_PACKAGING\_INSTALL\_PREFIX](https://cmake.org/cmake/help/v3.5/variable/CPACK_PACKAGING_INSTALL_PREFIX.html)
* [CPACK\_SET\_DESTDIR](https://cmake.org/cmake/help/v3.5/variable/CPACK_SET_DESTDIR.html)
* [CPACK\_WARN\_ON\_ABSOLUTE\_INSTALL\_DESTINATION](https://cmake.org/cmake/help/v3.5/variable/CPACK_WARN_ON_ABSOLUTE_INSTALL_DESTINATION.html)

## **1.5 opt[第三方大型软件或游戏，相当于D:/Software]**

* opt代表可选的。
* 包含从个别厂商的附加应用程序。
* 附加应用程序应该安装在/opt/或者/opt/的子目录下。
* 这里主要存放那些可选的程序。你想尝试最新的firefox测试版吗?那就装到/opt目录下吧，这样，当你尝试完，想删掉firefox的时候，你就可 以直接删除它，而不影响系统其他任何设置。安装到/opt目录下的程序，它所有的数据、库文件等等都是放在同个目录下面。

举个例子：

刚才装的测试版firefox，就可以装到/opt/firefox\_beta目录下，/opt/firefox\_beta目录下面就包含了运 行firefox所需要的所有文件、库、数据等等。要删除firefox的时候，你只需删除/opt/firefox\_beta目录即可，非常简单。

# 第二部分 管理相关

## **2.1 etc[配置文件]**

* 包含所有程序所需的配置文件。
* 也包含了用于启动/停止单个程序的启动和关闭shell脚本。
* 例如：/etc/resolv.conf、/etc/logrotate.conf
* 该目录下的文件由系统管理员来使用，普通用户对大部分文件有只读权限。

### /etc/profile

设置系统级别环境变量

### /etc/bashrc

设置系统级别bash相关特性

### /etc/issue

### /etc/redhat-release

### /etc/inittab

### /etc/fstab

### /etc/init.d

### /etc/X11**（X Window系统有关）**

### /etc/sysconfig**（与网络有关）**

### /etc/xinetd.d**修改配置文件之前记得备份。**

## **2.2 dev[设备文件]**

* 包含设备文件。
* 这些包括终端设备、USB或连接到系统的任何设备。
* 例如：/dev/tty1、/dev/usbmon0

## **2.3 home[用户目录]**

* 所有用户用home目录来存储他们的个人档案。  
  例如：/home/john、/home/nikita

### /home/.profile

## **2.4 var[变量文件]**

* var代表变量文件。
* 这个目录下可以找到内容可能增长的文件。

### **/var/cache**

应用程序的缓存文件。

### **/var/lib**

应用程序的信息、数据，如数据库的数据等都存放在该文件夹中。

### **/var/local**

/usr/local中程序的信息、数据。

### **/var/lock**

锁文件。

### **/var/log**

日志文件。

### **/var/opt**

/opt中程序的信息、数据。

### **/var/run**

正在执行着的程序信息。

### **/var/spool**

存放程序的假脱机数据。

### **/var/tmp**

临时数据。

## **2.5 srv[服务数据]**

* srv代表服务。
* 包含服务器特定服务相关的数据。  
  例如，/srv/cvs包含cvs相关的数据。

## **2.6 tmp[临时文件]**

* 包含系统和用户创建的临时文件。
* 当系统重新启动时，这个目录下的文件都将被删除。

## **2.7 mnt[挂载目录]**

* 临时安装目录，系统管理员可以挂载文件系统。

## **2.8 media[可移动媒体设备]**

* 用于挂载可移动设备的临时目录。  
  举例来说，挂载CD-ROM的/media/cdrom，挂载软盘驱动器的/media/floppy;

# 第三部分 操作系统

## **3.1 proc[进程信息]**

* 包含系统进程的相关信息。
* 这是一个虚拟的文件系统，包含有关正在运行的进程的信息。例如：/proc/{pid}目录中包含的与特定pid相关的信息。
* 这是一个虚拟的文件系统，系统资源以文本信息形式存在。例如：/proc/uptime

### **/proc/cmdline 文件**

这个文件给出了内核启动的命令行。它和用于进程的cmdline 项非常相似。

### **/proc/cpuinfo 文件**

这个文件提供了有关系统CPU 的多种信息。这些信息是从内核里对CPU 的测试代码中得到的。文件列出 了CPU 的普通型号 （386,486,586,686 等 ）, 以及能得到的更多特定信息 （ 制造商, 型号和版本 ） 。 文件还包含 了以bogomips 表示的处理器速度, 而且如果检测到CPU 的多种特性或者bug, 文件还会包含相应的标志。这 个文件的格式为：文件由多行构成, 每行包括一个域名称, 一个冒号和一个值。

### **/proc/devices 文件**

这个文件列出字符和块设备的主设备号, 以及分配到这些设备号的设备名称。

### **/proc/dma 文件**

这个文件列出由驱动程序保留的DMA 通道和保留它们的驱动程序名称 。casade 项供用于把次DMA控制器从主控制器分出的DMA 行所使用; 这一行不能用于其它用途。

### **/proc/filesystems 文件**

这个文件列出可供使用的文件系统类型, 一种类型一行 。 虽然它们通常是编入内核的文件系统类型, 但该文件还可以包含可加载的内核模块加入的其它文件系统类型。

### **/proc/interrupts 文件**

这个文件的每一行都有一个保留的中断 。 每行中的域有 ： 中断号, 本行中断的发生次数, 可能带有一个加号的域（SA\_INTERRUPT 标志设置）, 以及登记这个中断的驱动程序的名字。可以在安装新硬件前, 像查看/proc/dma 和/proc/ioports 一样用cat 命令手工查看手头的这个文件 。 这几个文件列出了当前投入使用的资源（但是不包括那些没有加载驱动程序的硬件所使用的资源 ） 。

/proc/ioports 文件

这个文件列出了诸如磁盘驱动器, 以太网卡和声卡设备等多种设备驱动程序登记的许多I/O 端口范围。

### **/proc/kcore 文件**

这个文件是系统的物理内存以core 文件格式保存的文件。例如,GDB 能用它考察内核的数据结构。它不是纯文本, 而是/proc 目录下为数不多的几个二进制格式的项之一。

### **/proc/kmsg 文件**

这个文件用于检索用printk 生成的内核消息。任何时刻只能有一个具有超级用户权限的进程可以读取这个文件。也可以用系统调用syslog 检索这些消息。通常使用工具dmesg 或守护进程klogd 检索这些消息。

### **/proc/ksyms 文件**

这个文件列出了已经登记的内核符号; 这些符号给出了变量或函数的地址 。 每行给出一个符号的地址, 符号名称以及登记这个符号的模块。程序ksyms,insmod 和kmod 使用这个文件。它还列出了正在运行的任务数, 总任务数和最后分配的PID 。

### **/proc/loadavg 文件**

这个文件给出以几个不同的时间间隔计算的系统平均负载, 这就如同uptime 命令显示的结果那样 。 前三个数字是平均负载 。 这是通过计算过去1 分钟,5 分钟,15 分钟里运行队列中的平均任务数得到的 。 随后是正在运行的任务数和总任务数。最后是上次使用的进程号。

### **/proc/locks 文件**

这个文件包含在打开的文件上的加锁信息。文件中的每一行描述了特定文件和文档上的加锁信息以及对文件施加的锁的类型。内核也可以需要时对文件施加强制性锁。

### **/proc/mdstat 文件**

这个文件包含了由md 设备驱动程序控制的RAID 设备信息。

### **/proc/meminfo 文件**

这个文件给出了内存状态的信息 。 它显示出系统中空闲内存, 已用物理内存和交换内存的总量 。 它还显示出内核使用的共享内存和缓冲区总量。这些信息的格式和free 命令显示的结果类似。

### **/proc/misc 文件**

这个文件报告用内核函数misc\_register 登记的设备驱动程序。

### **/proc/modules 文件**

这个文件给出可加载内核模块的信息 。lsmod 程序用这些信息显示有关模块的名称, 大小, 使用数目方面的信息。

### **/proc/mounts 文件**

这个文件以/etc/mtab 文件的格式给出当前系统所安装的文件系统信息。这个文件也能反映出任何手工安装从而在/etc/mtab 文件中没有包含的文件系统。

### **/proc/pci 文件**

这个文件给出PCI 设备的信息。用它可以方便地诊断PCI 问题。你可以从这个文件中检索到的信息包括诸如IDE 接口或USB 控制器这样的设备, 总线, 设备和功能编号, 设备延迟以及IRQ 编号。

### **/proc/stat 文件**

这个文件包含的信息有CPU 利用率, 磁盘, 内存页, 内存对换, 全部中断, 接触开关以及赏赐自举时间（自1970 年1 月1 日起的秒数 ） 。

### **/proc/uptime 文件**

这个文件给出自从上次系统自举以来的秒数, 以及其中有多少秒处于空闲 。 这主要供uptime 程序使用 。 比较这两个数字能够告诉你长期来看CPU 周期浪费的比例。

#### /proc/version 文件|正在运行内核的版本

这个文件只有一行内容, 说明正在运行的内核版本。可以用标准的编程方法进行分析获得所需的系统信息 。

cat /proc/version

### **/proc/net 子目录**

此目录下的文件描述或修改了联网代码的行为。可以通过使用 arp,netstat,route 和ipfwadm 命令设置或查询这些特殊文件中的许多文件。

### **/proc/scsi 子目录** 此目录下包含一个列出了所有检测到的SCSI 设备的文件, 并且为每种控制器驱动程序提供一个目录, 在这个目录下又为已安装的此种控制器的每个实例提供一个子目录。

### **/proc/sys 子目录**

在此目录下有许多子目录 。 此目录中的许多项都可以用来调整系统的性能 。 这个目录包含信息太多, 无法介绍全部。只在示例中展示目录下的一些文件。

## **3.2 sys**

* ysfs 虚拟文件系统提供了一种比 proc 更为理想的访问内核数据的途径
* 简介：  
  sysfs 是 Linux 内核中设计较新的一种虚拟的基于内存的文件系统，它的作用与 proc 有些类似，但除了与 proc 相同的具有查看和设定内核参数功能之外，还有为 Linux 统一设备模型作为管理之用。相比于 proc 文件系统，使用 sysfs 导出内核数据的方式更为统一，并且组织的方式更好，它的设计 proc 中吸取了很多教训。
* sysfs 与 /sys   
  sysfs 文件系统总是被挂载在 /sys 挂载点上。虽然在较早期的2.6内核系统上并没有规定 sysfs 的标准挂载位置，可以把 sysfs 挂载在任何位置，但较近的2.6内核修正了这一规则，要求 sysfs 总是挂载在 /sys 目录上；针对以前的 sysfs 挂载位置不固定或没有标准被挂载，有些程序从 /proc/mounts 中解析出 sysfs 是否被挂载以及具体的挂载点，这个步骤现在已经不需要了。请参考附录给出的 sysfs-rules.txt 文件链接。
* sysfs 与 proc   
  sysfs 与 proc 相比有很多优点，最重要的莫过于设计上的清晰。一个 proc 虚拟文件可能有内部格式，如 /proc/scsi/scsi ，它是可读可写的，(其文件权限被错误地标记为了 0444 ！，这是内核的一个BUG)，并且读写格式不一样，代表不同的操作，应用程序中读到了这个文件的内容一般还需要进行字符串解析，而在写入时需要先用字符串格式化按指定的格式写入字符串进行操作；相比而言， sysfs 的设计原则是一个属性文件只做一件事情， sysfs 属性文件一般只有一个值，直接读取或写入。整个 /proc/scsi 目录在2.6内核中已被标记为过时(LEGACY)，它的功能已经被相应的 /sys 属性文件所完全取代。新设计的内核机制应该尽量使用 sysfs 机制，而将 proc 保留给纯净的“进程文件系统”。

### **/sys/devices**

该目录下是全局设备结构体系，包含所有被发现的注册在各种总线上的各种物理设备。一般来说，所有的物理设备都按其在总线上的拓扑结构来显示，但有两个例外，即platform devices和system devices。 platform devices一般是挂在芯片内部的高速或者低速总线上的各种控制器和外设，它们能被CPU直接寻址；system devices不是外设，而是芯片内部的核心结构，比如CPU，timer等，它们一般没有相关的驱动，但是会有一些体系结构相关的代码来配置它们。

(sys/devices是内核对系统中所有设备的分层次表达模型，也是/sys文件系统管理设备的最重要的目录结构)

### **/sys/dev**

该目录下维护一个按照字符设备和块设备的主次号码(major:minor)链接到真是设备(/sys/devices)的符号链接文件。

### **/sys/class**

该目录下包含所有注册在kernel里面的设备类型，这是按照设备功能分类的设备模型，每个设备类型表达具有一种功能的设备。每个设备类型子目录下都是这种设备类型的各种具体设备的符号链接，这些链接指向/sys/devices/name下的具体设备。设备类型和设备并没有一一对应的关系，一个物理设备可能具备多种设备类型；一个设备类型只表达具有一种功能的设备，比如：系统所有输入设备都会出现在/sys/class/input之下，而不论它们是以何种总线连接到系统的。(/sys/class也是构成linux统一设备模型的一部分)

#### /sys/class/gpio

##### /sys/class/gpio/export

##### **/sys/class/gpio/unexport**

### **/sys/block**

该目录下的所有子目录代表着系统中当前被发现的所有块设备。按照功能来说放在/sys/class下会更合适，但由于历史遗留因素而一直存在于/sys/block，但从linux2.6.22内核开始这部分就已经标记为过去时，只有打开了CONFIG\_SYSFS\_DEPRECATED配置编译才会有这个目录存在，并且其中的内容在从linux2.6.26版本开始已经正式移到了/sys/class/block，旧的接口/sys/block为了向后兼容而保留存在，但其中的内容已经变为了指向它们在/sys/devices/中真实设备的符号链接文件。

### **/sys/bus**

该目录下的每个子目录都是kernel支持并且已经注册了的总线类型。这是内核设备按照总线类型分层放置的目录结构，/sys/devices中的所有设备都是连接于某种总线之下的，bus子目录下的每种具体总线之下可以找到每个具体设备的符号链接，

一般来说每个子目录(总线类型)下包含两个子目录，一个是devices，另一个是drivers；其中devices下是这个总线类型下的所有设备，这些设备都是符号链接，它们分别指向真正的设备(/sys/devices/name/下)；而drivers下是所有注册在这个总线上的驱动，每个driver子目录下 是一些可以观察和修改的driver参数。

(它也是构成linux统一设备模型的一部分)

### **/sys/fs**

按照设计，该目录使用来描述系统中所有的文件系统，包括文件系统本身和按照文件系统分类存放的已挂载点。

### **/sys/kernel**

这个目录下存放的是内核中所有可调整的参数

### **/sys/firmware**

该目录下包含对固件对象(firmware object)和属性进行操作和观察的接口，即这里是系统加载固件机制的对用户空间的接口.(关于固件有专用于固件加载的一套API)

### **/sys/hypervisor**

该目录是与虚拟化Xen相关的装置。(Xen是一个开放源代码的虚拟机监视器)

### **/sys/module**

该目录下有系统中所有的模块信息，不论这些模块是以内联(inlined)方式编译到内核映像文件中还是编译为外模块(.ko文件)，都可能出现在/sys/module中。即module目录下包含了所有的被载入kernel的模块。

### **/sys/power**

该目录是系统中的电源选项，对正在使用的power子系统的描述。这个目录下有几个属性文件可以用于控制整个机器的电源状态，如可以向其中写入控制命令让机器关机/重启等等。

## **3.3 boot[引导加载程序文件]**

* 包含引导加载程序相关的文件。
* 内核的initrd、vmlinux、grub文件位于/boot下。  
  例如：initrd.img-2.6.32-24-generic、vmlinuz-2.6.32-24-generic

## **3.4 root**

* 系统管理员root的目录

## **3.5 selinux**

# 第四部分 日志相关

## **4.1 logs**

* 记录日志

## **4.2 lost+found**

* 在文件系统修复时恢复的文件