**gn是什么?**

gn 存在的意义是为了生成 ninja,如果熟悉前端开发,二者关系很像 Sass和CSS的关系. 为什么会有gn,说是有个叫even的谷歌负责构建系统的工程师在使用传统的makefile构建chrome时觉得太麻烦,不高效,所以设计了一套更简单,更高效新的构建工具gn+ninja,然后就被广泛的使用了.

**gn语法和配置**

gn 有大量的内置变量和库函数,熟悉这些库函数基本就知道gn能干什么,gn的官方文档很齐全,前往 [gn参考手册](https://www.oschina.net/action/GoToLink?url=http%3A%2F%2Fweharmonyos.com%2Fgn%2Fdocs%2Freference.html)翻看查找 . gn 的语法简单,了解以下几点基本就能看懂gn代码.重点了解下函数的调用方式,和其他高级语言不太一样.

**字符串**

a = "mypath"

b = "$a/foo.cc" # b -> "mypath/foo.cc"

c = "foo${a}bar.cc" # c -> "foomypathbar.cc"

**列表**

a = [ "first" ]

a += [ "second" ] # [ "first", "second" ]

a += [ "third", "fourth" ] # [ "first", "second", "third", "fourth" ]

b = a + [ "fifth" ] # [ "first", "second", "third", "fourth", "fifth" ]

**条件语句**

if (is\_linux || (is\_win && target\_cpu == "x86")) {

sources -= [ "something.cc" ]

}else {

...

}

**循环**

foreach(i, mylist) {

print(i) # Note: i is a copy of each element, not a reference to it.

}

**函数调用**

print("hello, world")

assert(is\_win, "This should only be executed on Windows") # 如果is\_win为真,就打印后面的内容

static\_library("mylibrary") {

sources = [ "a.cc" ]

}

解读:

* print,assert,static\_library都是库函数,或者叫内置函数.
* static\_library的调用有点奇怪,它是个函数,函数内部会使用sources这个内置变量,sources = [ "a.cc" ]相当于先传参给这个函数的内置变量,并调用这个函数.学习ng得习惯这种语法方式,被大量的使用.

**模板 | Templates**

gn提供了很多内置函数,使用偏傻瓜式,若构建不复杂的系统,熟悉这些内置函数,选择填空就可以交作业了,如果想高阶点,想自己定义函数怎么办? 答案是模板,模板就是自定义函数.

#定义模板, 文件路径: //tools/idl\_compiler.gni, 后缀.gni 代表这是一个 gn import file

template("idl") { #自定义一个名称为 "idl"的函数

source\_set(target\_name) { #调用内置函数 source\_set

sources = invoker.sources #invoker为内置变量,含义为调用者内容 即:[ "a", "b" ]的内容

}

}

#如何使用模板, 用import,类似 C语言的 #include

import("//tools/idl\_compiler.gni")

idl("my\_interfaces") { #等同于调用 idl

sources = [ "a", "b" ] #给idl传参, 参数的接收方是 invoker.sources

}

明白了模板的使用,阅读鸿蒙gn代码就不会有太大的障碍.

**目标项 | Targets**

目标是构建图中的一个节点。它通常表示将生成某种可执行文件或库文件。整个构建是由一个个的目标组成. 目标包含:

action: 运行脚本以生成文件

executable: 生成可执行文件

group: 生成依赖关系组

shared\_library: 生成.dll或.so动态链接库

static\_library: 生成.lib或.a 静态链接库

...

**配置项 | Configs**

记录完成目标项所需的配置信息,例如:

config("myconfig") {#创建一个标签为`myconfig`的配置项

include\_dirs = [ "include/common" ]

defines = [ "ENABLE\_DOOM\_MELON" ]

}

executable("mything") {#生成可执行文件

configs = [ ":myconfig" ]#使用标签为`myconfig`的配置项来生成目标文件

}

**gn在鸿蒙中的使用**

有了以上基础铺垫,正式开始gn在openharomny中的使用.

**从哪开始**

在构建工具篇中已经说清楚了 hb的python部分有个工作任务是生成gn命令所需参数. gn生成ninja的命令是 gn gen ...

/home/tools/gn gen /home/openharmony/code-v1.1.1-LTS/out/hispark\_aries/ipcamera\_hispark\_aries \

--root=/home/openharmony/code-v1.1.1-LTS \

--dotfile=/home/openharmony/code-v1.1.1-LTS/build/lite/.gn \

--script-executable=python3 \

'--args=ohos\_build\_type="debug" \

ohos\_build\_compiler\_specified="clang" \

ohos\_build\_compiler\_dir="/home/tools/llvm" \

product\_path="/home/openharmony/code-v1.1.1-LTS/vendor/hisilicon/hispark\_aries" \

device\_path="/home/openharmony/code-v1.1.1-LTS/device/hisilicon/hispark\_aries/sdk\_liteos" \

ohos\_kernel\_type="liteos\_a" \

enable\_ohos\_appexecfwk\_feature\_ability = false \

ohos\_full\_compile=true'

**解读**

* root,dotfile,script-executable是gn内置的固定参数,一切从dotfile指向的文件开始.即build/lite/.gn 相当于main()函数的作用,
* 打开 [build/lite/.gn](https://gitee.com/weharmony/build_lite/blob/master/.gn)看下内容,只有两句,先配置好内部参数再工作.
* # The location of the build configuration file. #1.完成gn的配置工作
* buildconfig = "//build/lite/config/BUILDCONFIG.gn"
* # The source root location. #2.完成gn的编译工作
* root = "//build/lite"
* args为用户自定义的参数,它们将会在解析BUILD.gn,BUILDCONFIG.gn过程中被使用.

**BUILDCONFIG.gn | 构建配置项**

BUILDCONFIG.gn为BUILD.gn做准备,填充好编译所需的配置信息.即生成配置项 详细请查看 [build/lite/config/BUILDCONFIG.gn](https://gitee.com/weharmony/build_lite/blob/master/config/BUILDCONFIG.gn) 文件全部内容,本篇只贴出部分.

import("//build/lite/ohos\_var.gni")

import("${device\_path}/config.gni")

....

arch = "arm"

if (ohos\_kernel\_type == "liteos\_a") {

target\_triple = "$arch-liteos"

} else if (ohos\_kernel\_type == "linux") {

target\_triple = "$arch-linux-ohosmusl"

}

...

template("executable") { #生成可执行文件

executable(target\_name) {

forward\_variables\_from(invoker, Variables\_Executable)

if (!defined(invoker.deps)) {

deps = [ "//build/lite:prebuilts" ]

} else {

deps += [ "//build/lite:prebuilts" ]

}

if (defined(invoker.configs)) {

configs = []

configs += invoker.configs

}

}

}

set\_defaults("executable") {#设置目标类型的默认值

configs = default\_executable\_configs

configs += [ "//build/lite/config:board\_exe\_ld\_flags" ]

}

...

**解读**

* ${device\_path}为命令行参数,本篇为home/openharmony/code-v1.1.1-LTS/device/hisilicon/hispark\_aries/sdk\_liteos
* 查看[构建-配置](https://www.oschina.net/action/GoToLink?url=http%3A%2F%2Fweharmonyos.com%2Fgn%2Fdocs%2Flanguage.html%23build-configuration)内容,BUILDCONFIG主要任务就是对其中的内置变量赋值.例如:
  + 指定编译器 clang,
  + 如何生成可执行文件的方法等

**BUILD.gn | 启动构建**

查看 [build/lite/BUILD.gn](https://gitee.com/weharmony/build_lite/blob/master/BUILD.gn) 文件,文件注解较多.

#目的是要得到项目各个模块的编译入口

group("ohos") {

deps = []

if (ohos\_build\_target == "") {

# Step 1: Read product configuration profile.

# 第一步:读取配置文件product\_path的值来源于根目录的ohos\_config.json,如下,内容由 hb set 命令生成

# {

# "root\_path": "/home/openharmony",

# "board": "hispark\_aries",

# "kernel": "liteos\_a",

# "product": "ipcamera\_hispark\_aries",

# "product\_path": "/home/openharmony/vendor/hisilicon/hispark\_aries",

# "device\_path": "/home/openharmony/device/hisilicon/hispark\_aries/sdk\_liteos",

# "patch\_cache": null

#}

product\_cfg = read\_file("${product\_path}/config.json", "json")

# Step 2: Loop subsystems configured by product.

# 第二步:循环处理各自子系统,config.json中子系统部分格式如下hb

#"subsystems": [

# {

# "subsystem": "aafwk",

# "components": [

# { "component": "ability", "features":[ "enable\_ohos\_appexecfwk\_feature\_ability = false" ] }

# ]

# },

# ...

# {

# "subsystem": "distributed\_schedule",

# "components": [

# { "component": "system\_ability\_manager", "features":[] },

# { "component": "foundation", "features":[] },

# { "component": "distributed\_schedule", "features":[] }

# ]

# },

# {

# "subsystem": "kernel",

# "components": [

# { "component": "liteos\_a", "features":[] }

# ]

# },

#]

foreach(product\_configed\_subsystem, product\_cfg.subsystems) {#对子系统数组遍历操作

subsystem\_name = product\_configed\_subsystem.subsystem #读取一个子系统 aafwk,hiviewdfx,security ==

subsystem\_info = {

}

# Step 3: Read OS subsystems profile.

# 第三步: 读取各个子系统的配置文件

subsystem\_info =

read\_file("//build/lite/components/${subsystem\_name}.json", "json")

# Step 4: Loop components configured by product.

# 第四步: 循环读取子系统内各控件的配置信息

# 此处以内核为例://build/lite/components/kernel.json"

# "components": [

# {

# "component": "liteos\_a", # 组件名称

# "description": "liteos-a kernel", # 组件一句话功能描述

# "optional": "false", # 组件是否为最小系统必选

# "dirs": [ # 组件源码路径

# "kernel/liteos\_a"

# ],

# "targets": [ # 组件编译入口

# "//kernel/liteos\_a:kernel"

# ],

# "rom": "1.98MB", # 组件ROM值

# "ram": "", # 组件RAM估值

# "output": [ # 组件编译输出

# "liteos.bin"

# ],

# "adapted\_board": [ # 组件已适配的主板

# "hispark\_aries",

# "hispark\_taurus",

# "hi3518ev300",

# "hi3516dv300",

# ],

# "adapted\_kernel": [ "liteos\_a" ], # 组件已适配的内核

# "features": [], # 组件可配置的特性

# "deps": {

# "components": [], # 组件依赖的其他组件

# "third\_party": [ # 组件依赖的三方开源软件

# "FreeBSD",

# "musl",

# "zlib",

# "FatFs",

# "Linux\_Kernel",

# "lwip",

# "NuttX",

# "mtd-utils"

# ]

# }

# },

# ]

foreach(product\_configed\_component,

product\_configed\_subsystem.components) { #遍历项目控件数组

# Step 5: Check whether the component configured by product is exist.

# 第五步: 检查控件配置信息是否存在

component\_found = false #初始为不存在

foreach(system\_component, subsystem\_info.components) {#项目控件和子系统中的控件遍历对比

if (product\_configed\_component.component ==

system\_component.component) { #找到了liteos\_a

component\_found = true

}

}

#如果没找到的信息,则打印项目控件查找失败日志

assert(

component\_found,

"Component \"${product\_configed\_component.component}\" not found" +

", please check your product configuration.")

# Step 6: Loop OS components and check validity of product configuration.

# 第六步: 检查子系统控件的有效性并遍历控件组,处理各个控件

foreach(component, subsystem\_info.components) {

kernel\_valid = false #检查内核

board\_valid = false #检查开发板

# Step 6.1: Skip component which not configured by product.

if (component.component == product\_configed\_component.component) {

# Step 6.1.1: Loop OS components adapted kernel type.

foreach(component\_adapted\_kernel, component.adapted\_kernel) {

if (component\_adapted\_kernel == product\_cfg.kernel\_type &&

kernel\_valid == false) { #内核检测是否已适配

kernel\_valid = true

}

}

# 如果内核未适配,则打印未适配日志

assert(

kernel\_valid,

"Invalid component configed, ${subsystem\_name}:${product\_configed\_component.component} " + "not available for kernel: ${product\_cfg.kernel\_type}!")

# Step 6.1.2: Add valid component for compiling.

# 添加有效组件进行编译

foreach(component\_target, component.targets) {//遍历组件的编译入口

deps += [ component\_target ] #添加到编译列表中

}

}

}

}

}

# Step 7: Add device and product target by default.

# 第七步: 添加设备和项目的编译单元

# "product\_path": "/home/openharmony/vendor/hisilicon/hispark\_aries",

# "device\_path": "/home/openharmony/device/hisilicon/hispark\_aries/sdk\_liteos",

deps += [

"${device\_path}/../", #添加 //device/hisilicon/hispark\_aries 进入编译项

"${product\_path}" #添加 //vendor/hisilicon/hispark\_aries 进入编译项

]

} else {#编译指定的组件,例如 hb build -T targetA&&targetB

deps += string\_split(ohos\_build\_target, "&&")

}

}

**解读**

* 有三个概念贯彻整个鸿蒙系统,子系统(subsystems),组件(components),功能(features).理解它们的定位和特点是解读鸿蒙的关键所在.
* 先找到product\_path下的 配置文件 config.json,里面配置了项目所要使用的子系统和组件.
* 再遍历项目所使用的组件是否能再 //build/lite/components/\*.json组件集中能找到.
* 将找到的组件targets加入到编译列表deps中.targets指向了要编译的组件目录.例如内核组件时指向了:[//kernel/liteos\_a:kernel](https://gitee.com/weharmony/kernel_liteos_a_note/blob/master/BUILD.gn),
* import("//build/lite/config/component/lite\_component.gni") #组件模板函数
* import("//build/lite/config/subsystem/lite\_subsystem.gni") #子系统模板函数
* lite\_subsystem("kernel") {#编译内核子系统/组件入口
* subsystem\_components = []
* if (enable\_ohos\_kernel\_liteos\_a\_ext\_build == false) {
* subsystem\_components += [
* "//kernel/liteos\_a/kernel",
* "//kernel/liteos\_a/net",
* "//kernel/liteos\_a/lib",
* "//kernel/liteos\_a/compat",
* "//kernel/liteos\_a/fs",
* "//kernel/liteos\_a/arch:platform\_cpu",
* ]
* if (LOSCFG\_SHELL) {
* subsystem\_components += [ "//kernel/liteos\_a/shell" ]
* }
* } else {
* deps = [ ":make" ]
* }
* }

lite\_subsystem是个模板函数(自定义函数),再查看[lite\_subsystem.gni](https://gitee.com/weharmony/build_lite/blob/master/config/subsystem/lite_subsystem.gni)函数原型,它的目的只有一个填充 [deps](https://www.oschina.net/action/GoToLink?url=http%3A%2F%2Fweharmonyos.com%2Fgn%2Fdocs%2Freference.html%23var_deps),deps是私有链接依赖关系,最终会形成一颗依赖树.gn会根据这些依赖关系生成最终的.ninja文件.

# 定义一个子系统

# lite\_subsystem template模板定义了子系统中包含的所有模块

# 参数

# subsystem\_components (必须))

# [范围列表] 定义子系统的所有模块.

template("lite\_subsystem") {

assert(defined(invoker.subsystem\_components), "subsystem\_components in required.")

lite\_subsystem\_components = invoker.subsystem\_components

group(target\_name) {

deps = []

if(defined(invoker.deps)) {

deps += invoker.deps

}

# add subsystem packages

foreach(pkg\_label, lite\_subsystem\_components) {

deps += [ pkg\_label ]

}

}

}

**生成了哪些文件**

执行后gn gen 会生成如下文件和目录

turing@ubuntu:/home/openharmony/code-v1.1.1-LTS/out/hispark\_aries/ipcamera\_hispark\_aries$ ls

args.gn build.ninja build.ninja.d NOTICE\_FILE obj test\_info toolchain.ninja

build.ninja.d中记录依赖的 BUILD.gn文件路径

build.ninja: ../../../base/global/resmgr\_lite/frameworks/resmgr\_lite/BUILD.gn \

../../../base/hiviewdfx/hilog\_lite/frameworks/featured/BUILD.gn \

../../../base/hiviewdfx/hilog\_lite/services/apphilogcat/BUILD.gn \

....

gn根据这些组件的BUILD.gn在obj目录下对应生成了每个组件的.ninja文件.此处列出鸿蒙L1所有的 .ninja文件, 具体ninja是如何编译成最终的库和可执行文件的,将在后续篇中详细介绍其语法和应用.

turing@ubuntu:/home/openharmony/code-v1.1.1-LTS/out/hispark\_aries/ipcamera\_hispark\_aries/obj$ tree

├── base

│   ├── global

│   │   └── resmgr\_lite

│   │   └── frameworks

│   │   └── resmgr\_lite

│   │   └── global\_resmgr.ninja

│   ├── hiviewdfx

│   │   └── hilog\_lite

│   │   ├── frameworks

│   │   │   └── featured

│   │   │   ├── hilog\_shared.ninja

│   │   │   └── hilog\_static.ninja

│   │   └── services

│   │   ├── apphilogcat

│   │   │   ├── apphilogcat.ninja

│   │   │   └── apphilogcat\_static.ninja

│   │   └── hilogcat

│   │   ├── hilogcat.ninja

│   │   └── hilogcat\_static.ninja

│   ├── security

│   │   ├── appverify

│   │   │   └── interfaces

│   │   │   └── innerkits

│   │   │   └── appverify\_lite

│   │   │   ├── products

│   │   │   │   └── ipcamera

│   │   │   │   └── verify\_base.ninja

│   │   │   ├── unittest

│   │   │   │   └── app\_verify\_test.ninja

│   │   │   └── verify.ninja

│   │   ├── deviceauth

│   │   │   └── frameworks

│   │   │   └── deviceauth\_lite

│   │   │   └── source

│   │   │   └── hichainsdk.ninja

│   │   ├── huks

│   │   │   └── frameworks

│   │   │   └── huks\_lite

│   │   │   └── source

│   │   │   └── huks.ninja

│   │   └── permission

│   │   └── services

│   │   └── permission\_lite

│   │   ├── ipc\_auth

│   │   │   └── ipc\_auth\_target.ninja

│   │   ├── pms

│   │   │   └── pms\_target.ninja

│   │   ├── pms\_base

│   │   │   └── pms\_base.ninja

│   │   └── pms\_client

│   │   └── pms\_client.ninja

│   └── startup

│   ├── appspawn\_lite

│   │   └── services

│   │   ├── appspawn.ninja

│   │   └── test

│   │   └── unittest

│   │   └── common

│   │   └── appspawn\_test.ninja

│   ├── bootstrap\_lite

│   │   └── services

│   │   └── source

│   │   └── bootstrap.ninja

│   ├── init\_lite

│   │   └── services

│   │   ├── init.ninja

│   │   └── test

│   │   └── unittest

│   │   └── common

│   │   └── init\_test.ninja

│   └── syspara\_lite

│   └── frameworks

│   ├── parameter

│   │   └── src

│   │   └── sysparam.ninja

│   ├── token

│   │   └── token\_shared.ninja

│   └── unittest

│   └── parameter

│   └── ParameterTest.ninja

├── build

│   └── lite

│   └── config

│   └── component

│   ├── cJSON

│   │   ├── cjson\_shared.ninja

│   │   └── cjson\_static.ninja

│   ├── openssl

│   │   ├── openssl\_shared.ninja

│   │   └── openssl\_static.ninja

│   └── zlib

│   ├── zlib\_shared.ninja

│   └── zlib\_static.ninja

├── drivers

│   ├── adapter

│   │   └── uhdf

│   │   ├── manager

│   │   │   └── hdf\_core.ninja

│   │   ├── platform

│   │   │   └── hdf\_platform.ninja

│   │   ├── posix

│   │   │   └── hdf\_posix\_osal.ninja

│   │   └── test

│   │   └── unittest

│   │   ├── common

│   │   │   └── hdf\_test\_common.ninja

│   │   ├── config

│   │   │   └── hdf\_adapter\_uhdf\_test\_config.ninja

│   │   ├── manager

│   │   │   ├── hdf\_adapter\_uhdf\_test\_door.ninja

│   │   │   ├── hdf\_adapter\_uhdf\_test\_ioservice.ninja

│   │   │   ├── hdf\_adapter\_uhdf\_test\_manager.ninja

│   │   │   └── hdf\_adapter\_uhdf\_test\_sbuf.ninja

│   │   ├── osal

│   │   │   └── hdf\_adapter\_uhdf\_test\_osal.ninja

│   │   └── platform

│   │   └── hdf\_adapter\_uhdf\_test\_platform.ninja

│   └── peripheral

│   ├── input

│   │   └── hal

│   │   └── hdi\_input.ninja

│   └── wlan

│   ├── client

│   │   └── wifi\_driver\_client.ninja

│   ├── hal

│   │   └── wifi\_hal.ninja

│   └── test

│   ├── performance

│   │   └── hdf\_peripheral\_wlan\_test\_performance.ninja

│   └── unittest

│   └── hdf\_peripheral\_wlan\_test.ninja

├── foundation

│   ├── aafwk

│   │   └── aafwk\_lite

│   │   ├── frameworks

│   │   │   ├── ability\_lite

│   │   │   │   └── ability.ninja

│   │   │   ├── abilitymgr\_lite

│   │   │   │   └── abilitymanager.ninja

│   │   │   └── want\_lite

│   │   │   └── want.ninja

│   │   └── services

│   │   └── abilitymgr\_lite

│   │   ├── abilityms.ninja

│   │   ├── tools

│   │   │   └── aa.ninja

│   │   └── unittest

│   │   └── test\_lv0

│   │   └── page\_ability\_test

│   │   └── ability\_test\_pageAbilityTest\_lv0.ninja

│   ├── ai

│   │   └── engine

│   │   ├── services

│   │   │   ├── client

│   │   │   │   ├── ai\_client.ninja

│   │   │   │   ├── client\_executor

│   │   │   │   │   └── client\_executor.ninja

│   │   │   │   └── communication\_adapter

│   │   │   │   └── ai\_communication\_adapter.ninja

│   │   │   ├── common

│   │   │   │   ├── platform

│   │   │   │   │   ├── dl\_operation

│   │   │   │   │   │   └── dlOperation.ninja

│   │   │   │   │   ├── event

│   │   │   │   │   │   └── event.ninja

│   │   │   │   │   ├── lock

│   │   │   │   │   │   └── lock.ninja

│   │   │   │   │   ├── os\_wrapper

│   │   │   │   │   │   └── ipc

│   │   │   │   │   │   └── aie\_ipc.ninja

│   │   │   │   │   ├── semaphore

│   │   │   │   │   │   └── semaphore.ninja

│   │   │   │   │   ├── threadpool

│   │   │   │   │   │   └── threadpool.ninja

│   │   │   │   │   └── time

│   │   │   │   │   └── time.ninja

│   │   │   │   ├── protocol

│   │   │   │   │   └── data\_channel

│   │   │   │   │   └── data\_channel.ninja

│   │   │   │   └── utils

│   │   │   │   └── encdec

│   │   │   │   └── encdec.ninja

│   │   │   └── server

│   │   │   ├── ai\_server.ninja

│   │   │   ├── communication\_adapter

│   │   │   │   └── ai\_communication\_adapter.ninja

│   │   │   ├── plugin\_manager

│   │   │   │   └── plugin\_manager.ninja

│   │   │   └── server\_executor

│   │   │   └── server\_executor.ninja

│   │   └── test

│   │   ├── common

│   │   │   ├── ai\_test\_common.ninja

│   │   │   └── dl\_operation

│   │   │   └── dl\_operation\_so

│   │   │   └── dlOperationSo.ninja

│   │   ├── function

│   │   │   ├── ai\_test\_function.ninja

│   │   │   └── death\_callback

│   │   │   ├── testDeathCallbackLibrary.ninja

│   │   │   └── testDeathCallback.ninja

│   │   ├── performance

│   │   │   └── ai\_test\_performance\_unittest.ninja

│   │   └── sample

│   │   ├── asyncDemoPluginCode.ninja

│   │   ├── sample\_plugin\_1.ninja

│   │   ├── sample\_plugin\_2.ninja

│   │   └── syncDemoPluginCode.ninja

│   ├── appexecfwk

│   │   └── appexecfwk\_lite

│   │   ├── frameworks

│   │   │   └── bundle\_lite

│   │   │   └── bundle.ninja

│   │   └── services

│   │   └── bundlemgr\_lite

│   │   ├── bundle\_daemon

│   │   │   └── bundle\_daemon.ninja

│   │   ├── bundlems.ninja

│   │   └── tools

│   │   └── bm.ninja

│   ├── communication

│   │   ├── ipc\_lite

│   │   │   └── liteipc\_adapter.ninja

│   │   └── softbus\_lite

│   │   └── softbus\_lite.ninja

│   ├── distributedschedule

│   │   ├── dmsfwk\_lite

│   │   │   ├── dmslite.ninja

│   │   │   └── moduletest

│   │   │   └── dtbschedmgr\_lite

│   │   │   └── distributed\_schedule\_test\_dms.ninja

│   │   ├── safwk\_lite

│   │   │   └── foundation.ninja

│   │   └── samgr\_lite

│   │   ├── communication

│   │   │   └── broadcast

│   │   │   └── broadcast.ninja

│   │   ├── samgr

│   │   │   ├── adapter

│   │   │   │   └── samgr\_adapter.ninja

│   │   │   ├── samgr.ninja

│   │   │   └── source

│   │   │   └── samgr\_source.ninja

│   │   ├── samgr\_client

│   │   │   └── client.ninja

│   │   ├── samgr\_endpoint

│   │   │   ├── endpoint\_source.ninja

│   │   │   └── store\_source.ninja

│   │   └── samgr\_server

│   │   └── server.ninja

│   ├── graphic

│   │   ├── surface

│   │   │   ├── surface.ninja

│   │   │   └── test

│   │   │   └── lite\_surface\_unittest.ninja

│   │   ├── ui

│   │   │   └── ui.ninja

│   │   ├── utils

│   │   │   ├── graphic\_hals.ninja

│   │   │   ├── graphic\_utils.ninja

│   │   │   └── test

│   │   │   ├── graphic\_test\_color.ninja

│   │   │   ├── graphic\_test\_container.ninja

│   │   │   ├── graphic\_test\_geometry2d.ninja

│   │   │   ├── graphic\_test\_math.ninja

│   │   │   └── graphic\_test\_style.ninja

│   │   └── wms

│   │   ├── wms\_client.ninja

│   │   └── wms\_server.ninja

│   └── multimedia

│   ├── audio\_lite

│   │   └── frameworks

│   │   └── audio\_capturer\_lite.ninja

│   ├── camera\_lite

│   │   └── frameworks

│   │   └── camera\_lite.ninja

│   ├── media\_lite

│   │   ├── frameworks

│   │   │   ├── player\_lite

│   │   │   │   └── player\_lite.ninja

│   │   │   └── recorder\_lite

│   │   │   └── recorder\_lite.ninja

│   │   ├── interfaces

│   │   │   └── kits

│   │   │   └── player\_lite

│   │   │   └── js

│   │   │   └── builtin

│   │   │   └── audio\_lite\_api.ninja

│   │   └── services

│   │   └── media\_server.ninja

│   └── utils

│   └── lite

│   └── media\_common.ninja

├── test

│   ├── developertest

│   │   ├── examples

│   │   │   └── lite

│   │   │   └── cxx\_demo

│   │   │   └── test

│   │   │   └── unittest

│   │   │   └── common

│   │   │   └── CalcSubTest.ninja

│   │   └── third\_party

│   │   └── lib

│   │   └── cpp

│   │   ├── gtest\_main.ninja

│   │   └── gtest.ninja

│   └── xts

│   ├── acts

│   │   ├── aafwk\_lite

│   │   │   └── ability\_posix

│   │   │   └── module\_ActsAbilityMgrTest.ninja

│   │   ├── ai\_lite

│   │   │   └── ai\_engine\_posix

│   │   │   └── base

│   │   │   ├── module\_ActsAiEngineTest.ninja

│   │   │   └── src

│   │   │   └── sample

│   │   │   ├── asyncDemoPluginCode.ninja

│   │   │   ├── sample\_plugin\_1\_sync.ninja

│   │   │   ├── sample\_plugin\_2\_async.ninja

│   │   │   └── syncDemoPluginCode.ninja

│   │   ├── appexecfwk\_lite

│   │   │   └── bundle\_mgr\_posix

│   │   │   └── module\_ActsBundleMgrTest.ninja

│   │   ├── communication\_lite

│   │   │   ├── lwip\_posix

│   │   │   │   └── module\_ActsLwipTest.ninja

│   │   │   └── softbus\_posix

│   │   │   └── module\_ActsSoftBusTest.ninja

│   │   ├── distributed\_schedule\_lite

│   │   │   └── samgr\_posix

│   │   │   └── module\_ActsSamgrTest.ninja

│   │   ├── graphic\_lite

│   │   │   ├── graphic\_utils

│   │   │   │   ├── a

│   │   │   │   │   └── module\_ActsUiInterfaceTest1.ninja

│   │   │   │   ├── color\_posix

│   │   │   │   │   └── module\_ActsColorTest.ninja

│   │   │   │   ├── geometry2d\_posix

│   │   │   │   │   └── module\_ActsGeometyr2dTest.ninja

│   │   │   │   ├── graphic\_math\_posix

│   │   │   │   │   └── module\_ActsGraphicMathTest.ninja

│   │   │   │   ├── heap\_base\_posix

│   │   │   │   │   └── module\_ActsHeapBaseTest.ninja

│   │   │   │   ├── list\_posix

│   │   │   │   │   └── module\_ActsListTest.ninja

│   │   │   │   ├── mem\_api\_posix

│   │   │   │   │   └── module\_ActsGraphMemApiTest.ninja

│   │   │   │   ├── rect\_posix

│   │   │   │   │   └── module\_ActsRectTest.ninja

│   │   │   │   ├── transform\_posix

│   │   │   │   │   └── module\_ActsTransformTest.ninja

│   │   │   │   └── version\_posix

│   │   │   │   └── module\_ActsGraphVersionTest.ninja

│   │   │   ├── surface

│   │   │   │   └── surface\_posix

│   │   │   │   └── module\_ActsSurfaceTest.ninja

│   │   │   └── ui

│   │   │   ├── a

│   │   │   │   └── module\_ActsUiInterfaceTest.ninja

│   │   │   ├── animator\_posix

│   │   │   │   └── module\_ActsAnimatorTest.ninja

│   │   │   ├── easing\_equation\_posix

│   │   │   │   └── module\_ActsEasingEquationTest.ninja

│   │   │   ├── events\_posix

│   │   │   │   └── module\_ActsEventsTest.ninja

│   │   │   ├── flexlayout\_posix

│   │   │   │   └── module\_ActsFlexlaoutTest.ninja

│   │   │   ├── gridlayout\_posix

│   │   │   │   └── module\_ActsGridLayoutTest.ninja

│   │   │   ├── image\_posix

│   │   │   │   └── module\_ActsImageTest.ninja

│   │   │   ├── interpolation\_posix

│   │   │   │   └── module\_ActsInterpoliationTest.ninja

│   │   │   ├── layout\_posix

│   │   │   │   └── module\_ActsLayoutTest.ninja

│   │   │   ├── listlayout\_posix

│   │   │   │   └── module\_ActsListlayoutTest.ninja

│   │   │   ├── screen\_posix

│   │   │   │   └── module\_ActsScreenTest.ninja

│   │   │   ├── style\_posix

│   │   │   │   └── module\_ActsStyleTest.ninja

│   │   │   ├── theme\_posix

│   │   │   │   └── module\_ActsThemeTest.ninja

│   │   │   ├── ui\_abstract\_progress\_posix

│   │   │   │   └── module\_ActsUIAbstractProgressTest.ninja

│   │   │   ├── ui\_analog\_clock\_posix

│   │   │   │   └── module\_ActsUIAnalogClockTest.ninja

│   │   │   ├── uianimator\_posix

│   │   │   │   └── module\_ActsUIAnimatorTest.ninja

│   │   │   ├── ui\_arc\_lable\_posix

│   │   │   │   └── module\_ActsUIArcLabelTest.ninja

│   │   │   ├── ui\_axis\_posix

│   │   │   │   └── module\_ActsUIAxisTest.ninja

│   │   │   ├── ui\_box\_porgress\_posix

│   │   │   │   └── module\_ActsUIBoxProgressTest.ninja

│   │   │   ├── ui\_button\_posix

│   │   │   │   └── module\_ActsUIButtonTest.ninja

│   │   │   ├── ui\_canvas\_posix

│   │   │   │   └── module\_ActsUICanvasTest.ninja

│   │   │   ├── ui\_chart\_posix

│   │   │   │   └── module\_ActsUIChartTest.ninja

│   │   │   ├── ui\_checbox\_posix

│   │   │   │   └── module\_ActsUICheckboxTest.ninja

│   │   │   ├── ui\_circle\_progress\_posix

│   │   │   │   └── module\_ActsUICircleProgressTest.ninja

│   │   │   ├── ui\_digital\_clock\_posix

│   │   │   │   └── module\_ActsUIDigitalClockTest.ninja

│   │   │   ├── ui\_image\_animator\_posix

│   │   │   │   └── module\_ActsUIImageAnimatorTest.ninja

│   │   │   ├── ui\_image\_posix

│   │   │   │   └── module\_ActsUIImageTest.ninja

│   │   │   ├── ui\_label\_button\_posix

│   │   │   │   └── module\_ActsUILabelButtonTest.ninja

│   │   │   ├── ui\_label\_posix

│   │   │   │   └── module\_ActsUILabelTest.ninja

│   │   │   ├── ui\_list\_posix

│   │   │   │   └── module\_ActsUIListTest.ninja

│   │   │   ├── ui\_picker\_posix

│   │   │   │   └── module\_ActsUIPickerTest.ninja

│   │   │   ├── ui\_radio\_button\_posix

│   │   │   │   └── module\_ActsUIRadioButtonTest.ninja

│   │   │   ├── ui\_repeat\_button\_posix

│   │   │   │   └── module\_ActsUIRepeatButtonTest.ninja

│   │   │   ├── ui\_screenshot\_posix

│   │   │   │   └── module\_ActsUIScreenshotTest.ninja

│   │   │   ├── ui\_scroll\_view\_posix

│   │   │   │   └── module\_ActsUIScrollViewTest.ninja

│   │   │   ├── ui\_slider\_posix

│   │   │   │   └── module\_ActsUISliderTest.ninja

│   │   │   ├── ui\_surface\_view\_posix

│   │   │   │   └── module\_ActsUISurfaceViewTest.ninja

│   │   │   ├── ui\_swipe\_view\_posix

│   │   │   │   └── module\_ActsUISwipeViewTest.ninja

│   │   │   ├── ui\_text\_posix

│   │   │   │   └── module\_ActsUITextTest.ninja

│   │   │   ├── ui\_texture\_mapper\_posix

│   │   │   │   └── module\_ActsUITextureMapperTest.ninja

│   │   │   ├── ui\_time\_picker\_posix

│   │   │   │   └── module\_ActsUITimePickerTest.ninja

│   │   │   ├── ui\_toggle\_button\_posix

│   │   │   │   └── module\_ActsUIToggleButtonTest.ninja

│   │   │   ├── ui\_view\_group\_posix

│   │   │   │   └── module\_ActsUIViewGroupTest.ninja

│   │   │   └── ui\_view\_posix

│   │   │   └── module\_ActsUIViewTest.ninja

│   │   ├── hiviewdfx\_lite

│   │   │   └── hilog\_posix

│   │   │   └── module\_ActsHilogTest.ninja

│   │   ├── kernel\_lite

│   │   │   ├── dyload\_posix

│   │   │   │   └── module\_ActsDyloadTest.ninja

│   │   │   ├── fs\_posix

│   │   │   │   ├── jffs

│   │   │   │   │   └── module\_ActsJFFS2Test.ninja

│   │   │   │   ├── nfs

│   │   │   │   │   └── module\_ActsNFSTest.ninja

│   │   │   │   ├── vfat

│   │   │   │   │   └── module\_ActsVFATTest.ninja

│   │   │   │   └── vfat\_storage

│   │   │   │   └── module\_ActsVFATstorageTest.ninja

│   │   │   ├── futex\_posix

│   │   │   │   └── module\_ActsFutexApiTest.ninja

│   │   │   ├── io\_posix

│   │   │   │   └── module\_ActsIoApiTest.ninja

│   │   │   ├── ipc\_posix

│   │   │   │   ├── message\_queue

│   │   │   │   │   └── module\_ActsIpcMqTest.ninja

│   │   │   │   ├── pipe\_fifo

│   │   │   │   │   └── module\_ActsIpcPipeTest.ninja

│   │   │   │   ├── semaphore

│   │   │   │   │   └── module\_ActsIpcSemTest.ninja

│   │   │   │   ├── shared\_memory

│   │   │   │   │   └── module\_ActsIpcShmTest.ninja

│   │   │   │   └── signal

│   │   │   │   └── module\_ActsIpcSignalTest.ninja

│   │   │   ├── math\_posix

│   │   │   │   ├── complexTest.ninja

│   │   │   │   └── module\_ActsMathApiTest.ninja

│   │   │   ├── mem\_posix

│   │   │   │   └── module\_ActsMemApiTest.ninja

│   │   │   ├── net\_posix

│   │   │   │   └── module\_ActsNetTest.ninja

│   │   │   ├── process\_posix

│   │   │   │   └── module\_ActsProcessApiTest.ninja

│   │   │   ├── sched\_posix

│   │   │   │   └── module\_ActsSchedApiTest.ninja

│   │   │   ├── sys\_posix

│   │   │   │   └── module\_ActsSysApiTest.ninja

│   │   │   ├── time\_posix

│   │   │   │   └── module\_ActsTimeApiTest.ninja

│   │   │   ├── util\_posix

│   │   │   │   └── module\_ActsUtilApiTest.ninja

│   │   │   └── utils

│   │   │   ├── libfs.ninja

│   │   │   ├── libmt\_utils.ninja

│   │   │   └── libutils.ninja

│   │   ├── multimedia\_lite

│   │   │   └── media\_lite\_posix

│   │   │   └── recorder\_native

│   │   │   └── module\_ActsMediaRecorderTest.ninja

│   │   ├── security\_lite

│   │   │   ├── datahuks\_posix

│   │   │   │   └── module\_ActsSecurityDataTest.ninja

│   │   │   └── permission\_posix

│   │   │   ├── capability

│   │   │   │   ├── capability\_shared.ninja

│   │   │   │   ├── jffs

│   │   │   │   │   └── module\_ActsJFFS2CapabilityTest.ninja

│   │   │   │   └── vfat

│   │   │   │   └── module\_ActsVFATCapabilityTest.ninja

│   │   │   ├── dac

│   │   │   │   ├── jffs

│   │   │   │   │   └── module\_ActsJFFS2DACTest.ninja

│   │   │   │   └── vfat

│   │   │   │   └── module\_ActsVFATDACTest.ninja

│   │   │   └── pms

│   │   │   └── module\_ActsPMSTest.ninja

│   │   ├── startup\_lite

│   │   │   ├── bootstrap\_posix

│   │   │   │   └── module\_ActsBootstrapTest.ninja

│   │   │   └── syspara\_posix

│   │   │   └── module\_ActsParameterTest.ninja

│   │   └── utils\_lite

│   │   └── kv\_store\_posix

│   │   └── module\_ActsKvStoreTest.ninja

│   └── tools

│   └── lite

│   ├── hcpptest

│   │   ├── gmock\_main.ninja

│   │   ├── gmock.ninja

│   │   ├── hcpptest\_main.ninja

│   │   └── hcpptest.ninja

│   └── others

│   └── query

│   └── query.ninja

├── third\_party

│   ├── bounds\_checking\_function

│   │   ├── libsec\_shared.ninja

│   │   └── libsec\_static.ninja

│   ├── freetype

│   │   └── freetype.ninja

│   ├── giflib

│   │   └── libgif.ninja

│   ├── iniparser

│   │   └── iniparser.ninja

│   ├── libjpeg

│   │   └── libjpeg.ninja

│   ├── libpng

│   │   └── libpng.ninja

│   ├── mbedtls

│   │   ├── mbedtls\_gt.ninja

│   │   ├── mbedtls\_shared.ninja

│   │   └── mbedtls\_static.ninja

│   └── qrcodegen

│   └── qrcodegen.ninja

├── utils

│   └── native

│   └── lite

│   ├── kv\_store

│   │   └── src

│   │   └── utils\_kv\_store.ninja

│   └── os\_dump

│   └── os\_dump.ninja

└── vendor

└── hisilicon

└── hispark\_aries

└── hals

├── security

│   └── permission\_lite

│   └── hal\_pms.ninja

└── utils

├── sys\_param

│   └── hal\_sysparam.ninja

└── token

└── haltoken\_shared.ninja