德赛西威

介绍NewBrush整个工程的目录结构以及编译和运行说明

NewBrush编译运行

[ NewBrush开发前准备 3](#_Toc82527945)

[ 相关环境配置 3](#_Toc82527946)

[ NewBrush数据库权限申请 3](#_Toc82527947)

[ NewBrush的目录结构 4](#_Toc82527948)

[ NewBrush的整个目录结构包含源码和编译配置脚本 4](#_Toc82527949)

[ Newbrush在各平台上的编译运行 5](#_Toc82527950)

[ Win32 5](#_Toc82527951)

[ 下载uit-newbrush 5](#_Toc82527952)

[ 切换到项目分支 5](#_Toc82527953)

[ 编译 5](#_Toc82527954)

[ 查看输出物 5](#_Toc82527955)

[ 运行 6](#_Toc82527956)

[ Android 8](#_Toc82527957)

[ 下载和切换分支与Win32一致不再赘述 8](#_Toc82527958)

[ 编译 8](#_Toc82527959)

[ 查看输出物 9](#_Toc82527960)

[ 运行 9](#_Toc82527961)

[ QNX 10](#_Toc82527962)

[ 下载uit-newbrush到/work 10](#_Toc82527963)

[ 切换分支与Win32一致不再赘述 11](#_Toc82527964)

[ 下载QNX SDK到/work/sdk下 11](#_Toc82527965)

[ 编译 11](#_Toc82527966)

[ 查看输出物 11](#_Toc82527967)

[ 运行 11](#_Toc82527968)

# NewBrush开发前准备

## 相关环境配置

参见[“工具和环境配置“文档](1.工具和环境配置.docx)

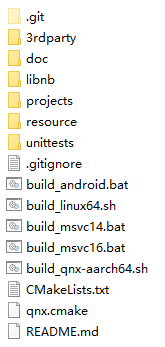
## NewBrush数据库权限申请

在scm-gitlab提issue进行NewBrush数据库权限的申请

数据库地址：<http://scm-gitlab/uit/nb/uit-newbrush>

# NewBrush的目录结构

## NewBrush的整个目录结构包含源码和编译配置脚本



* .git:存放git数据库的记录文件的文件夹
* 3rdparty：存放第三方库相关头文件及各平台下的库文件的文件夹
* doc：存放NewBrush相关的一些文档文件夹
* libnb：存放NewBrush的核心源码文件夹
* projects：存放基于NewBrush进行开发的UI项目
* resource：存放基于NewBrush进行开发的哥哥UI项目的资源文件
* unittests：存放NewBrush开发过程中进行单元测试的源码
* . gitignore:记录git 忽略的目录或文件
* build\_android.bat:Android编译脚本文件
* build\_msvc14.bat：Windows下VS2015编译脚本文件
* build\_msvc16.bat：Windows下VS2019编译脚本文件
* build\_qnx-aarch64.sh：QNX编译脚本文件
* CMakeLists.txt、qnx.cmake：用Cmake进行编译时的文件

# Newbrush在各平台上的编译运行

## Win32

* 下载uit-newbrush

（假设当前目录为D:/,则下载之后的目录结构为D：/uit-newbrush）

git clone [git@scm-gitlab:uit/nb/uit-newbrush.git](mailto:git@scm-gitlab:uit/nb/uit-newbrush.git)

* 切换到项目分支

cd uit-newbrush

git checkout –b xxx(local) xxx(remote)

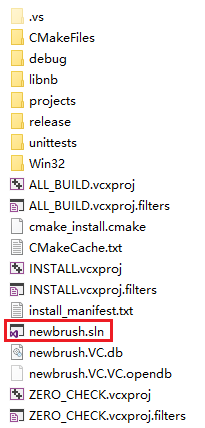
* 编译

build\_msvc14.bat

* 查看输出物



build：存放各平台项目相关工程文件，其中Win32文件夹下的目录结构如下图所示：



newbrush.sln是打开整个VS解决方案的文件，从这里进去打开整个VS解决方案，访问

和修改整个newbrush源码。

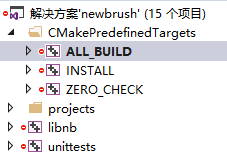
dist：存放各平台的可执行文件和相关资源，其中Win32文件夹下的目录结构如下图所

示：

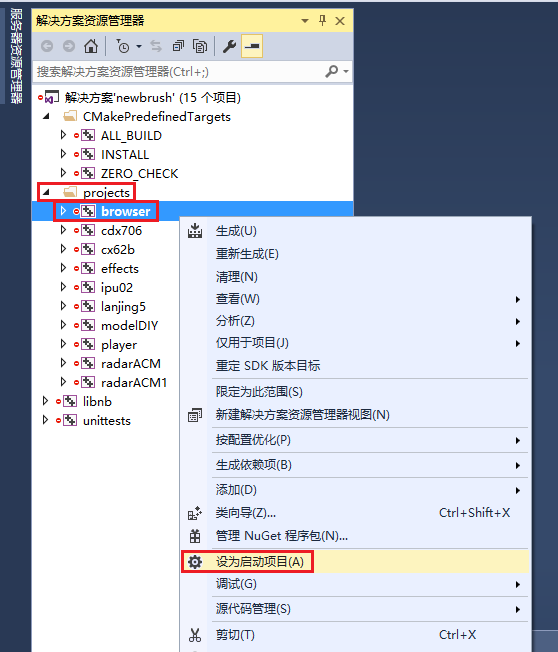


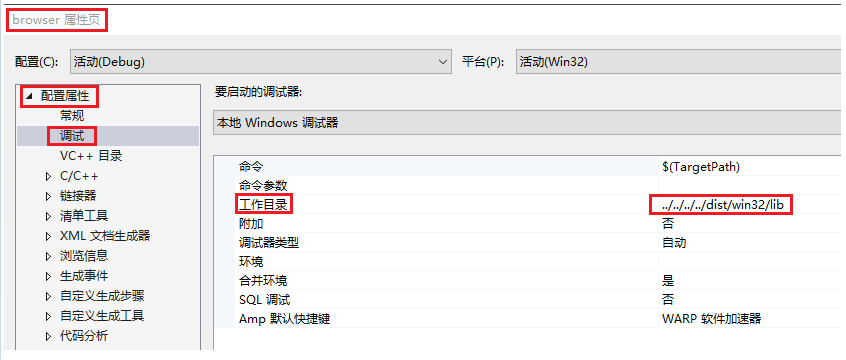
* 运行

进入VS项目，如下图所示：

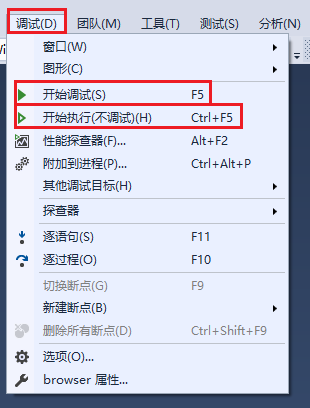


* 设置启动项目，如下图所示：



* 设置工作目录，如下图所示：

* 调试运行/直接运行（不调试）



## Android

* 下载和切换分支与Win32一致不再赘述
* 编译

（生成和安装Android APP依赖的so库和其余资源文件）

* 查看build\_android.bat文件，看Android相关工具路径是否正确

如果与本地电脑上安装的路径或版本不一致，请下载相应版本并存放到文件所示

路径或者更改该文件上的内容，使其与本地电脑上的一些配置匹配，如注意下图红框所示位置：



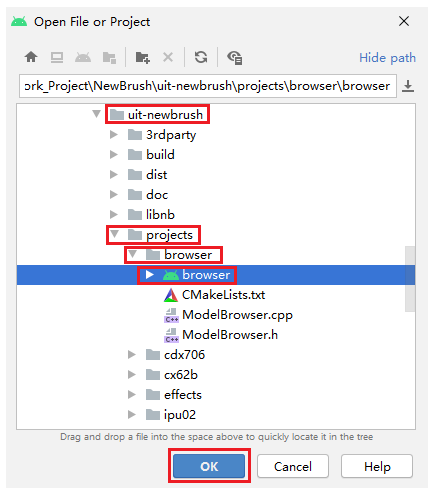
* 双击运行build\_android.bat
* 查看输出物

与Win32一样，会生成临时文件存放在build文件加里面，而完整的插件包含所有

依赖的so库、android java包以及使用到的图片资源等，它们在编译后自动输出到dist/android下。

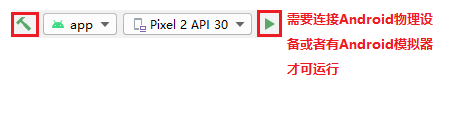
* 运行
* 通过Android Studio启动相应的Android项目，其中Android项目所在相对

文件路径如下图所示：



* 编译运行Android APP（上一步的编译只是为了生成和安装Andoird工程以来

的so库和java包而已），编译运行Android APP,如下图所示：



## QNX

* 下载uit-newbrush到/work

(假设当前目录为/work，那么直接运行下列命令)

git clone [git@scm-gitlab:uit/nb/uit-newbrush.git](mailto:git@scm-gitlab:uit/nb/uit-newbrush.git)

* 切换分支与Win32一致不再赘述
* 下载QNX SDK到/work/sdk下

（假设当前目录为/work/sdk,那么直接运行下列命令）

git clone git@scm-gitlab:sdk/sv-g6sh-qnx-system-sdk.git

* 编译

在work/uit-newbrush 目录下执行build\_qnx-aarch64.sh

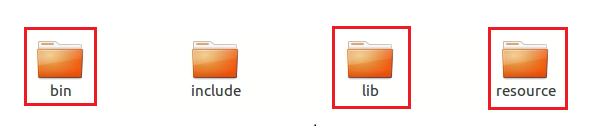
* 查看输出物

与Win32一样，会生成临时文件存放在build文件加里面，而完整的插件包含所

有依赖的so库以及使用到的图片资源等，它们在编译后自动输出到dist/qnx-aarch64下。

* 运行
* 把dist下的bin、lib和resource文件夹及其下的文件拷贝到机器上的

/emmc/newbrush目录下，如下图所示：



* 进入/emmc/newbrush /bin目录，执行下列命令

export LD\_LIBRARY\_PATH=../lib:$ LD\_LIBRARY\_PATH

./ browser