# 简陋的平台打包助手

版本：V1.1.2

更新日期：23.04.25

V1.1.2 更新内容

优化使用方式。

V0.0.6 更新内容

\* 双击退出按钮退出游戏

\* 隐私协议确认弹窗全屏化

\* Oppo内游戏全屏化。

V0.0.5 更新内容：

\* 【android-oppo】补充遗漏的构建项 OPPO\_APP\_KEY

\* 【android-mi】补充遗漏的构建项 MI\_APP\_KEY

\* 插件同步更新至 1.0.9。

\* 同步更新 插件构建参数说明.doc

V0.0.4 更新内容：

+ 增加对VIVO、小米、TapTap、Oppo的banner广告、插屏广告的支持

+ 增加 插件构建参数说明.doc

+ 现在插件打包时会自动更新项目的 Gradle 版本号配置和换国内源，移除相关文档说明

+ 增加对项目 Gradle.property 编码类型的调整的说明

+ 增加对 Gradle.SDK 版本要求的说明

V0.0.3 更新内容：

\* 安卓构建选项增加 隐私声明URL 和 用户协议URL

\* plat 增加接口 vibrate

\* plat vivo/oppo/xiaomi快游戏 接入banenr、插屏广告 （暂时只测试 Oppo下的横屏）

V0.0.2 更新内容：

+ 插件配置面板

+ 支持横屏

+ 【oppo】支持banner、插屏

+ plat接口变化

对应支持的 cc3平台版本：3.5.2,3.7.1

SVN地址：https://desktop-dvdsv79/svn/cgdr168/trunk/PlatHelper

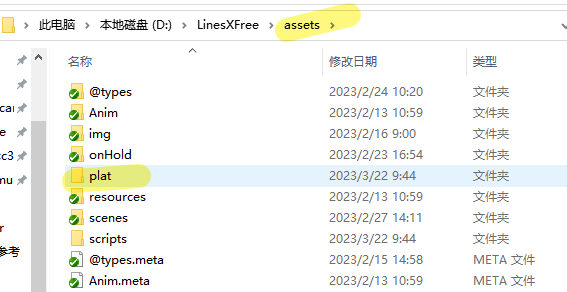
#### 文件结构



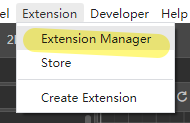
plat 脚本代码

extensions 插件

#### 安装及使用

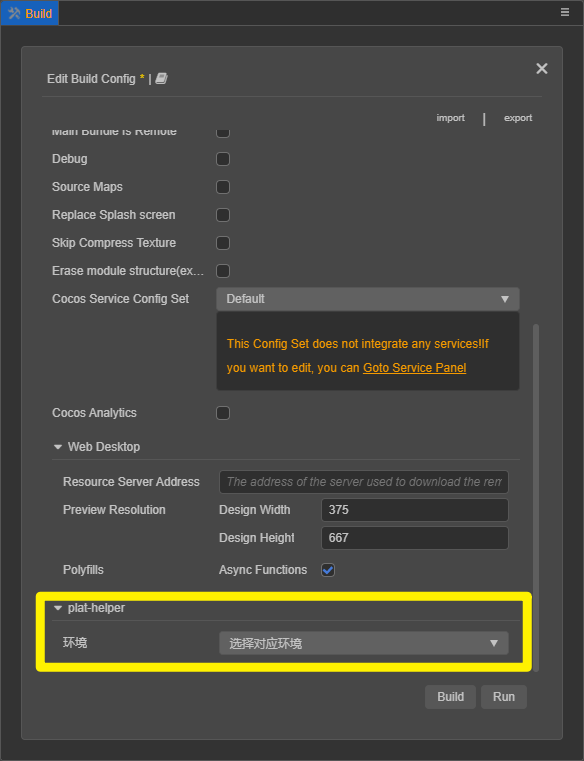


1. **将 plat 移动到cc项目的asset 下。然后根据报错处理相关代码。**



1. **安装插件。插件地址：**





插件安装完成后，在构建界面中，当目标平台为 Android 时将提供会有额外选项。

请按要求填写。

具体填写规则参考下面这个附件



1. **通过插件面板创建 AdCfg.ts （流程见附件视频）。AdCfg将用于配置广告位Id**



#### TS中使用

##### 必接

1. 在游戏入口调用

Plat.inst.init(...your init args...);

Plat.inst.setLoadingProgress(0);

1. 在确定玩家同意 用户协议/隐私声明 后调用

Plat.inst.setLoadingProgress(33);

Plat.inst.checkPlatReady(() => {

    Plat.inst.setLoadingProgress(66);

    Plat.inst.login(() => {

        Plat.inst.loadingComplete();

    }, uid);

});

**checkPlatReady**

等待原生平台原生代码初始化完毕。

**login**

处理“登录”及“防沉迷”

如果没有账户系统，uid可参考如下代码生成

let uid = sys.localStorage.getItem('NOW')

if (!uid) {

    uid = Date.now().toString() + "\_" + (Math.random() \* 100 << 0);

    sys.localStorage.setItem('NOW', uid);

}

1. 退出游戏时统一调用：

Plat.inst.endGame();

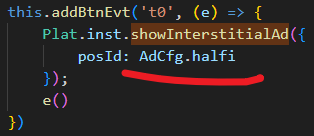
用法介绍

**AdCfg.ts 介绍**

配置时：

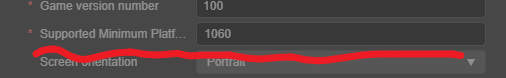


使用：



AdCfg.halfi 的实际值在不同的平台上将根据配置调整。

**小游戏打包**



#### 最小平台版本号推荐

|  |  |
| --- | --- |
| Oppo | >=1060 |
| Xiaomi | >=1056 |
| Vivo | >=1090 |
|  |  |

#### 安卓打包

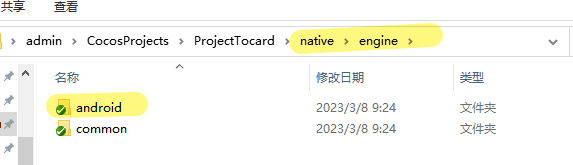
环境搭建



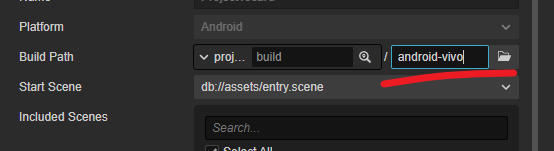
进入 [安装配置原生环境 · Cocos Creator](https://docs.cocos.com/creator/3.5/manual/zh/editor/publish/setup-native-development.html" \l "android-%E5%B9%B3%E5%8F%B0%E7%9B%B8%E5%85%B3%E4%BE%9D%E8%B5%96)，完成 Android平台相关依赖的安装

**首次打包**

下面以首次打vivo安卓包为例



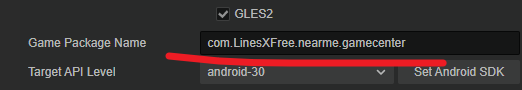
1. 如果存在需要删除 native/engine/android
2. 新建打包配置



**BuildPath**

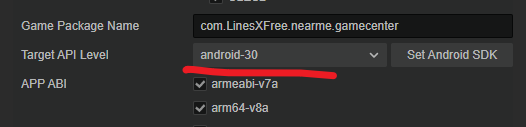
BuildPath 目录名命名为 android-vivo

|  |  |
| --- | --- |
| 平台 | BuildPath目录名 |
| Vivo | android-vivo |
| Taptap | android-taptap |
| 小米 | android-mi |
| Oppo | android-oppo |
| 应用宝 | android-yyb |



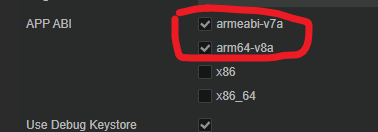
**包名**

请务必正确填写对应平台的管理后台中的包名



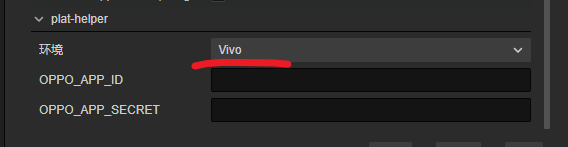
**TargetAPILevel 选择**

|  |  |
| --- | --- |
| vivo | 28 |
| Oppo | 30 |
| 小米 | 28 |
| Taptap | 28 |



**APP ABI**

APP ABI 勾上 armeabi-v7a 和 arm64-v8a

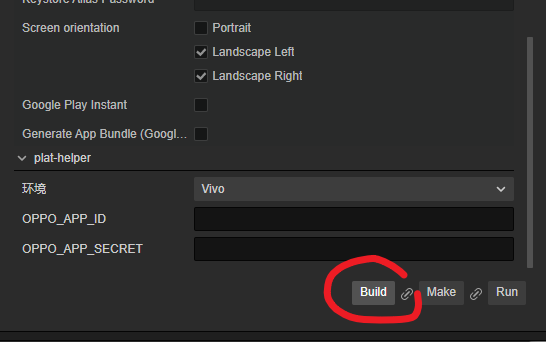


**Plat-helper**

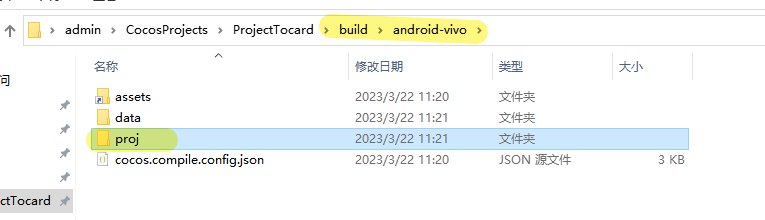
环境选择 Vivo

其它与Vivo相关的参数请对应填写。

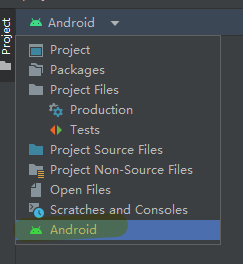
1. 构建



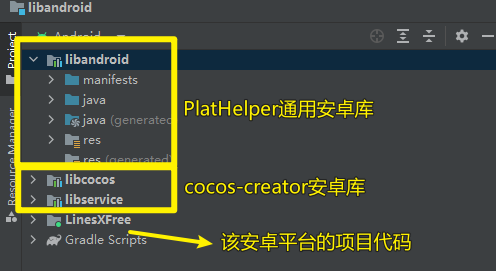
1. 等待构建结束。



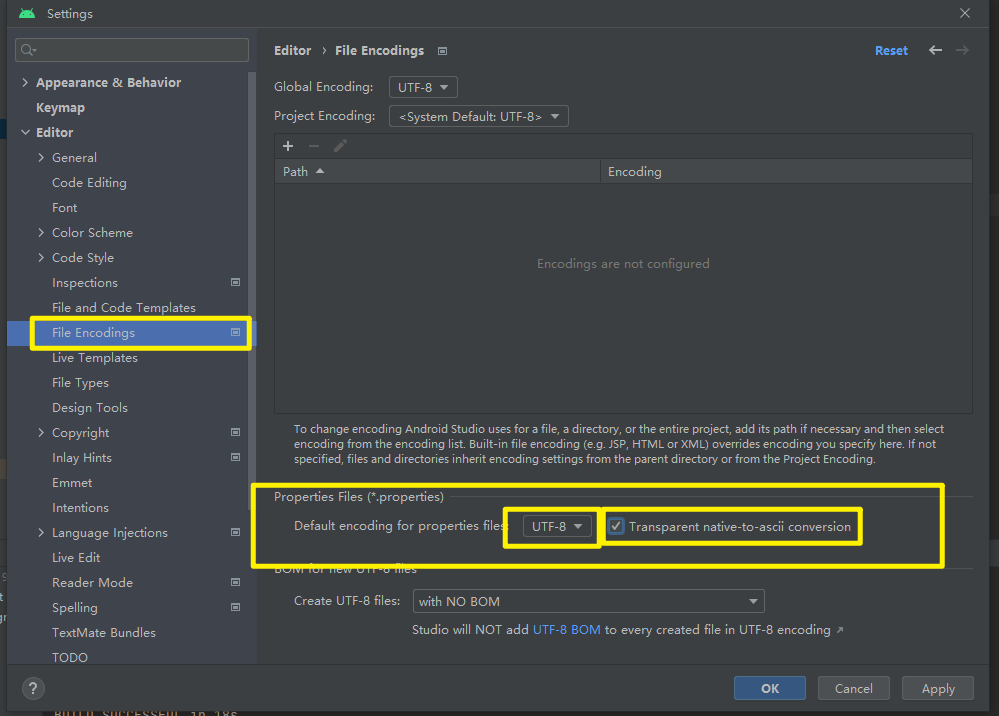
1. 用Android-studio 打开安卓项目



切换Project模式到 Android，如果没有这个选项，那么你可能项目地址选错了。



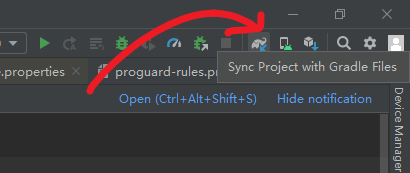
项目结构如上



5.5. 按图修改编码类型

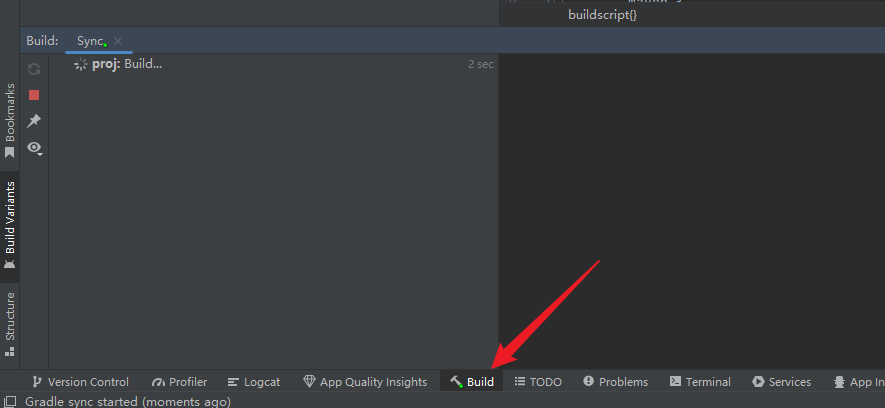
5.5. 按图修改编码类型

5.5. 按图修改编码类型

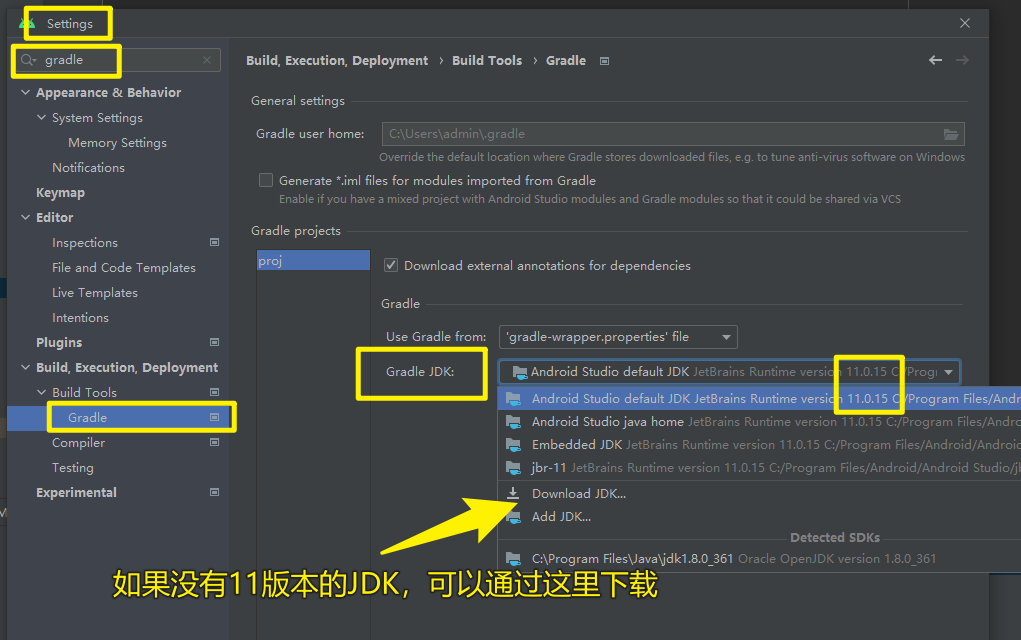


1. GradleSync 同步安卓包。

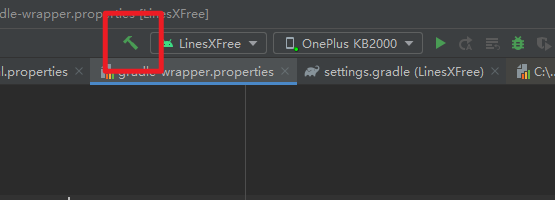
点击这个大象。



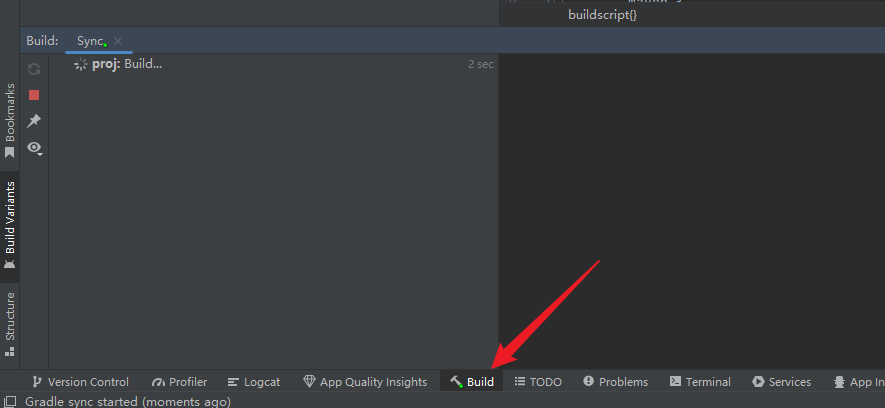
构建日志会显示在下面板的 Build 中。



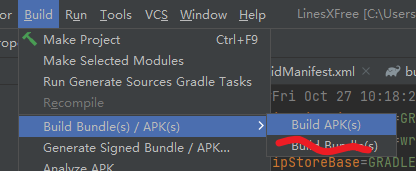
\* 期间如果报错，请确保当前 Gradle 使用的JDK版本为11。



1. Make 构建。GradleSync 如果成功就可以尝试 Make。



构建日志会显示在下面板的 Build 中。



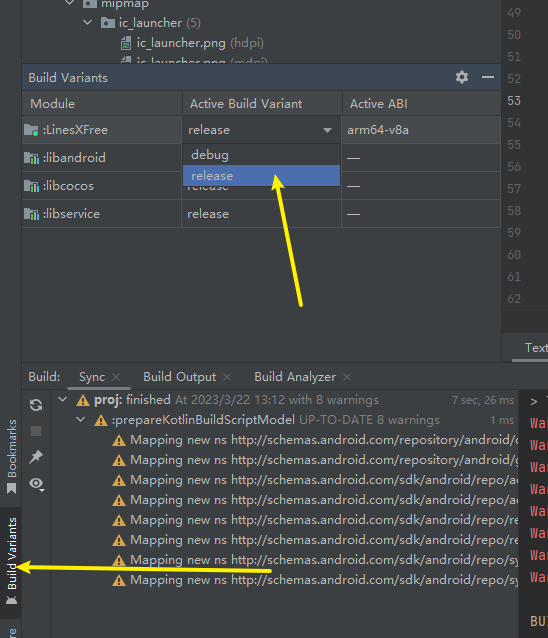
1. 构建 APK。Make如果成功，那么就可以构建APK了。

**Build Variants 变体**

\* 如果你正在调试代码，那么建议你切换 Variant 为debug，这样构建的时候会快一点。注意四个Module都要改为 debug。

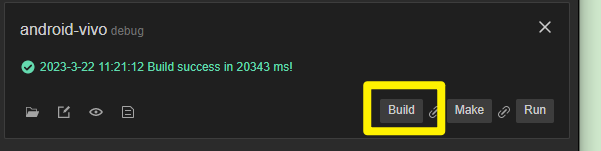
\* 如果你打算出正式包，请务必切换为 release 。

\* TapTap包只能在release下调试。



**后续打包流程**

1. Build



1. GradleSync
2. 按你希望的进行Make/Run/BuildApk

Q: 如何修改应用名称和应用图标

A：修改 native/engine 下平台项目文件夹的 res文件夹中的内容。

