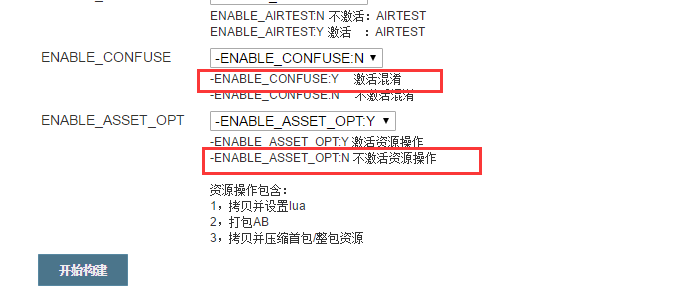
# 客户端混淆

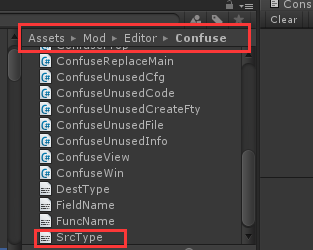
## 使用条件

1. 目前仅ios有此需求
2. 仅建立在已发包的基础之上
3. 不做logo/loading/闪屏等处理
4. Jenkins发布参数选项，选择激活混淆和不激活资源操作

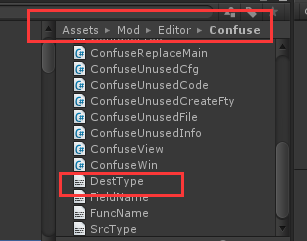


## 混淆内容

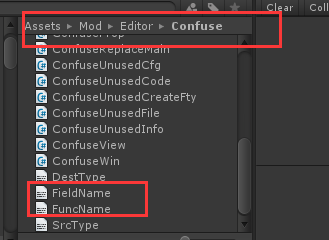
1. 无用资源混淆
   1. ，数量占比20%以上
   2. ，大小占比20%以上
   3. ，文件md5每次唯一性
   4. ，文件大小大概率不一致
   5. ，文件名称每次不一致
   6. ，文件内容二进制格式
   7. ，文件后缀名:xml/json/txt/bytes
   8. ，每次混淆文件数量+2
   9. ，文件名称根据字符排序，打乱CodeResources中顺序对比
2. 已有代码重命名
   1. ，此操作很危险，可能重命名需要和lua交互的字符串；当出现此问题时，需要将此字符串在关键字中屏蔽或者屏蔽此功能
   2. ，在配置文件中，定义了所有可以混淆的类型：



* 1. ，在配置文件中定义了将要混淆的类型,此类型(英文)必须是有意义的，数量越多越好



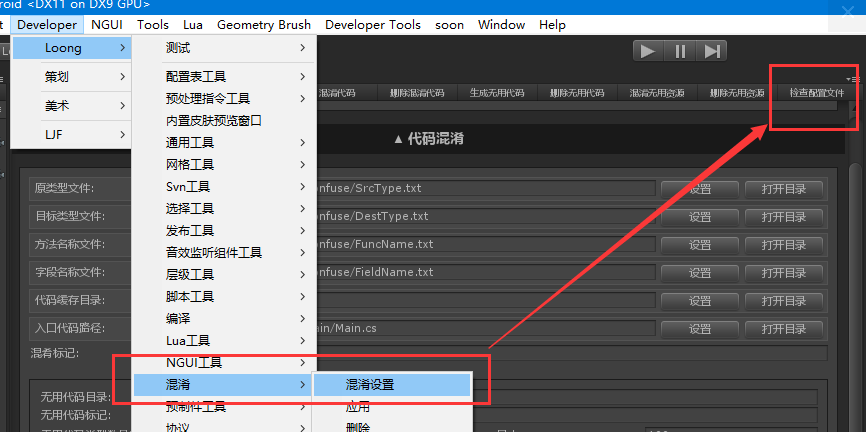
1. 无用代码生成
   1. ，无用类型数量100-300
   2. ，每个类型中属性数量为1-10
   3. ，每个类型中无用方法数量为1-5
   4. ，属性类型随机选择
   5. ，类型的基类至少有3种可选
   6. ，可读写属性随机选择
   7. ，访问等级随机选择
   8. ，至少一个公开等级方法
   9. ，方法体使用所有参数
   10. ，方法的返回值随机选择
   11. ，方法体内部使用+-\*/等操作；有字符串时进行字符串分割排序
   12. ，每种类型的默认值进行随机定义
   13. ，字段名称和方法名称的配置，越多越好：



1. 入口代码插入
   1. ，在main.cs脚本中关键位置插入
   2. ，插入代码从生成的无用代码中选择
   3. ，根据不同基类生成不同的初始化代码
   4. ，至少调用一个公开方法

## 配置校验

1. 混淆的配置文件内容越丰富，发生碰撞的概率越小
2. 使用如下菜单检查：



当出现非法内容时，控制台会有红色输出