Bugly (Android) SDK 快速接入向导

```
Bugly(Android )SDK 快速接入向导
一、库文件导入
方式1. 自动导入(推荐)
方式2. 手动导入

二、参数配置
三、代码调用
四、测试
接入成功
```

本文引导您最快地接入并可以使用Bugly,更多Bugly使用技巧请参看高级使用指南。同时我们在压缩包中提供了Eclipse和Android Studio的Demo工程。

一、库文件导入

方式1. 自动导入(推荐)

如果您使用Gradle编译Apk,我们强烈推荐您使用自动接入方式配置库文件和符号表插件。 在项目的buid.gradle文件的dependencies(buildscript部分)中添加符号表插件:

```
classpath 'com.tencent.bugly:symtabfileuploader:<version>' //其中<version>替换为当前最新符号表版
本号1.0.6
```

```
⊘ app × ⊘ BuglyDemoAndroidStudio
🖊 ⋤ BuglyDemoAndroidStudio (D:\Projects\Android\BuglyDemoA
 ▶ 🗀 .gradle
                                                                     repositories {
 ▼ 🛅 app
                                                                          jcenter()
    ▶ □ build
                                                                     dependencies {
    ▼ 🗖 src
                                                                                       com.android.tools.build:gradle:1.2.3
      ▶ □ androidTest
                                                                          classpath 'com.tencent.bugly:symtabfileuploader:1.0.6'
// NOTE: Do not place your application dependencies here; they belong
         ▶ □ java
         ▶ ☐ res
            AndroidManifest.xml
      gitignore
                                                                allprojects {
      I app.iml
                                                                     repositories {
      build.gradle
                                                                          jcenter()
      proguard-rules.pro
 ▶ □ gradle
    gitignore
    gradlew
    gradlew.bat
    local.properties
    constraint settings.gradle
  📫 External Libraries
```

在module的buid.gradle文件的顶部添加依赖和属性:

```
apply plugin: 'bugly' //添加Bugly符号表插件

dependencies {
    compile 'com.tencent.bugly:crashreport:<version>' //其中<version>替换为当前最新bugly版本号
1.2.3.8
}

bugly {
    appId = "your AppId" //注册时分配的App ID
    appKey = "your AppKey" //注册时分配的App Key
}
```

```
□ BuglyDemoAndroidStudio (D:\Projects\Android\BuglyDemoA
                                                             apply plugin: 'com.android.application'
                                                             apply plugin: 'bugly'
▼ 🛅 арр
   ▶ 🗀 build
                                                                  defaultConfig {
          ▼ 🗖 demo.bugly.tencent.com.buglydemo
               © ७ MainActivity
               © • MainApplication
          🙍 AndroidManifest.xml
    build.gradle
     proguard-rules.pro
▶ □ gradle
                                                                 compile fileTree(dir: 'libs', include: ['*.jar'])
   agitignore 🗎
                                                                 compile
   build.gradle
   a gradle.properties
                                                             bugly {
   gradlew
                                                                 appId = '<AppId>' //必需配置:您的AppId
appKey = '<AppKey>' //必需配置:您的AppKey
   gradlew.bat
   local.properties
                                                                   execute = true   //可选配置:是否生成符号表并自动上传,默认为true
upload = true   //可选配置:是否自动上传符号表开关,默认为true
   settings.gradle
External Libraries
```

至此库和自动符号表工具已经集成完成了。后续更新Bugly时,只需修改配置的版本号即可。

方式2. 手动导入

如果您不采用上述自动导入方式,也可以手动接入Bugly。

下载Bugly

请先下载Bugly的**SDK包**。

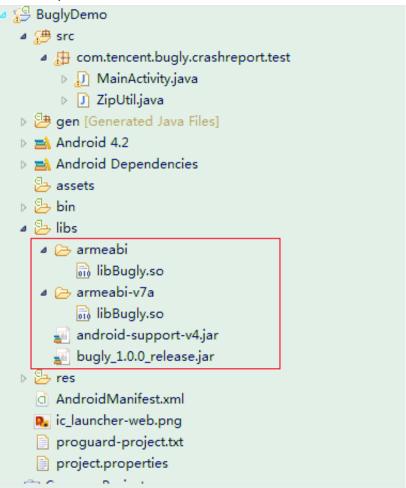
如果您的工程有Native(C++代码)代码,请继续下载NDK动态库和符号表工具(纯Java工程可跳过)。

下载地址: http://bugly.qq.com/whitebook

导入文件

如果您的Android工程是纯Java代码,只用导入Jar文件。如果含有Native代码,需一并导入so文件。

导入Eclipse



导入AndroidStudio

为了使您的堆栈可读性更高,对于纯Java代码项目,只需上传混淆后生成的Mapping.txt文件即可;对于含so的Native项目,除了Mapping.txt,还需要通过so手动提取Symbol符号表文件。具体配置方法请见:《更多配置(Android)和FAQ》

二、参数配置

在AndroidManifest.xml中添加权限:

```
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.READ_LOGS" />
```

避免混淆Bugly,请在Proguard混淆文件中配置:

```
keep public class com.tencent.bugly.**{*;}
```

三、代码调用

Bugly会为您检测使用环境并自动完成配置。请将以下CrashReport代码复制到项目Application类onCreate()中:

```
CrashReport.initCrashReport(appContext, "注册时申请的APPID", false);
```

四、测试

现在您可以制造一个Crash,来体验Bugly的能力了。在Bugly的初始化代码后,调用

```
CrashReport.testJavaCrash();
```

执行到这段代码时会发生一个Crash, 此时Logcat的TAG=CrashReportInfo中输出为:

现在您已经可以在网页上看到刚才的异常详情了(延迟一般在10s以内)。

接入成功

欢迎加入Bugly的大家庭!我们将忠实记录您App的每一次异常,尽力提供最详尽可靠的异常信息!如果您使用手动导入库文件方式接入Bugly,或者有其他自定义配置需求,请继续阅读《Bugly(Android)更多配置和FAQ》。

同时欢迎关注我们的公众号



在这里:

有Bugly在移动端的功能扩展 有腾讯大牛们的文章分享,和Bugly团队的干货 对Bugly的吐槽与赞美,也可以在这里与我们面对面