

SDK部署-Cordova方式

1、添加依赖

在 android 目录下的 build.gradle 文件中添加以下内容。

```
apply plugin: 'newlens'

buildscript {
    ext.tingyun_sdk_version = '2.13.9' //听云SDK版本
    ext.tingyun_ndk_version = '2.0.0' //听云NDK版本
    repositories {
        maven { url
"http://nexus2.tingyun.com/nexus/content/repositories/snapshots/" }
        }
        dependencies {
            classpath "com.networkbench.newlens.agent.android:agent-gradle-
plugin:$tingyun_sdk_version"
        }
    }

    allprojects {
        repositories {
            maven { url
"http://nexus2.tingyun.com/nexus/content/repositories/snapshots/" }
        }
    }

    dependencies {
        compile
        "com.networkbench.newlens.agent.android:nbs.newlens.agent:$tingyun_sdk_version"
        compile
        "com.networkbench.newlens.agent.android2:nbs.newlens.nativecrash:$tingyun_ndk_ve
rsion"
    }
}
```

在打开 CordovaLib 目录下的 build.gradle 文件中添加以下内容

```
dependencies {
    provided
    "com.networkbench.newlens.agent.android:nbs.newlens.agent:$tingyun_sdk_version"
}
```

2、配置权限

在AndroidManifest.xml文件中增加以下的权限

```

<!--必要权限，用以与服务端交互-->
<uses-permission android:name="android.permission.INTERNET"/>
<!--非必要权限，用以获取当前设备的网络状态和WiFi状态，如：2G、3G、4G、WiFi，建议添加-->
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<!--非必要权限，用以获取 targetSdkVersion 29 及以上 Android 10 设备的网络状态-->
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<!--非必要权限，用以使用「可视化操作命名功能」-->
<uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW"/>
<uses-permission android:name="android.permission.SYSTEM_OVERLAY_WINDOW"/>

```

可视化操作命名需要在 AndroidManifest.xml 文件的启动 Activity 增加 scheme 配置。
示例如下：

```

<activity android:name=".MainActivity">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
    <!--请添加这里的整个intent-filter区块，并确保其中只有一个data字段-->
    <intent-filter>
        <data android:scheme="tingyun.xxxx" />
        <!--请将 scheme 中的“tingyun.xxxx”替换为听云报表设置页面中的 URL
Scheme-->
        <action android:name="android.intent.action.VIEW"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <category android:name="android.intent.category.BROWSABLE"/>
    </intent-filter>
    <!--请添加这里的整个intent-filter区块，并确保其中只有一个data字段-->
</activity>

```

3、初始化SDK

在 “Application” 中的 onCreate() 方法初始化Android SDK。

```

NBSAppAgent.setLicenseKey("AppKey").startInApplication(this.getApplicationContext());
//Appkey请从官网获取

```

4、采集 WebVie数据配置

采集腾讯 X5 Webview 配置参见《高级功能》。

采集WebView数据需调用setWebViewClient方法，org.apache.cordova.engine.SystemWebView的setWebViewClient()已经设置，如没有该类或未调用该方法，请添加如下内容

```
webview.setWebViewClient(new WebViewClient(){});
```

采集WebView数据需在WebChromeClient的onProgressChanged函数中调用
NBSWebChromeClient.initJSMonitor(view, newProgress);

可在org.apache.cordova.engine.SystemWebChromeClient中重写onProgressChanged(), 代码示例如下：

```
@Override
public void onProgressChanged(WebView view, int newProgress) {
    com.networkbench.agent.impl.instrumentation.NBSWebChromeClient.initJSMonitor(view, newProgress);
}
```

5、配置混淆

在 proguard 混淆配置文件中增加以下内容，以免 tingyun SDK 不可用。

```
# ProGuard configurations for NetworkBench Lens
-keep class com.networkbench.** { *; }
-dontwarn com.networkbench.**
-keepattributes Exceptions, Signature, InnerClasses
# End NetworkBench Lens
```

若需要保留行号信息，请在 proguard.cfg 中添加以下内容。

```
-keepattributes SourceFile,LineNumberTable
```

6、使用Cordova命令打包编译

```
cordova build
```

7、嵌码验证

1、数据收集服务器校验。

2、嵌码完成后可通过“LogCat”查看听云SDK日志输出结果，用以进行数据收集服务器校验TAG为NBSAgent，标准日志输出结果如下所示：

```
NBSAgent start
NBSAgent enabled
NBSAgent V "TingYun_Version" //TingYun_Version 为当前SDK的版本号
connect success
```

3、数据功能完整性校验。

嵌码完成后可通过“LogCat”查看听云SDK日志输出结果，用以进行数据功能完整性校验TAG为TingYun，标准日志输出结果如下所示：

```
D/TingYun: Network Switch is true
D/TingYun: UI Switch is true
D/TingYun: Crash switch is true
D/TingYun: webView switch is true
D/TingYun: ANR monitor switch is true
D/TingYun: UserAction Switch is true
D/TingYun: cdnSwitch Switch is true
```

