Frontia Android 版

用户手册

发布日期： 2013年10月31日

百度开发者中心

目录

第1章 简介

Frontia 是一个完善的移动端 SDK。它集成了第三方账号登录、推送、个人数据存储、应用数据存储等模块，为开发者省去了自行开发和维护类似云服务的麻烦。Frontia 支持多种移动端（可到 http://developer.baidu.com/frontia/sdk 下载），Frontia Android 版是它的 Android 版本。

Frontia Android 版的完整下载包解压后，目录结构如下：

文件	描述
demo	存放 demo 的 eclipse 工程。直接导入 eclipse 后即可使用。
libs	Baidu-Frontia-Android-1.0.0.jar 是 Frontia Android 版类库。/armeabi/libpush-socket.so、/mips/libpush-socket.so 和 /x86/libpush-socket.so 是推送模块需要的动态库。
docs	Frontia Android SDK 接口文档。
Frontia_manual.pdf	使用手册。

第2章 运行环境

Frontia Android 版适用于 Android 2.2 及以上的系统版本。

第3章 使用前须知

3.1 开发准备

从 http://developer.android.com/sdk/index.html 下载适合自己平台的 android 开发工具包。解压后，里面带有一个配置好的 eclipse。

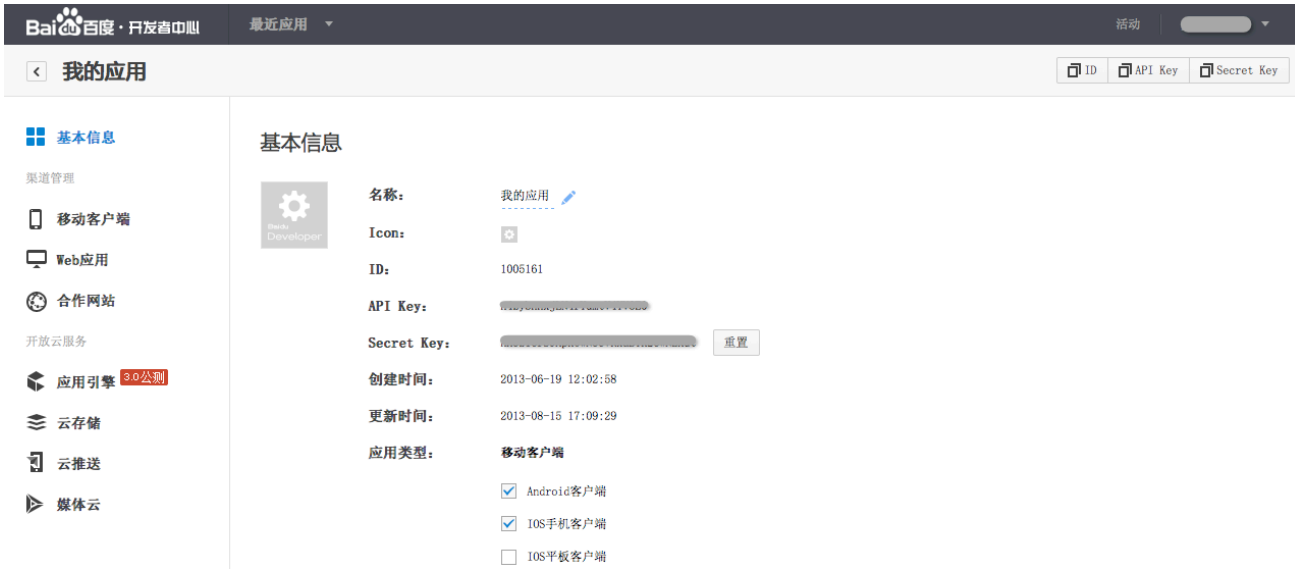
3.2 申请应用

1. 登录注册成为开发者，登录入口在 developer.baidu.com 首页右上方。
2. 申请应用，获得应用的 Api key。申请应用入口，在应用列表页（http://developer.baidu.com/console）点击“创建应用”按钮

   1）登录后顶部导航点击“新版管理控制台”到达应用列表页

2）点击“创建应用”进入创建页面，填写应用信息，并点击“创建”，则应用创建成功。

3）创建应用成功后，可到该应用下，点击左侧导航“基础信息”，查看 API Key 和 Secret Key



3.3 个人数据存储模块的配置

要使用 Frontia 的个人数据存储模块需要开通应用的 PCS API 权限。具体位置是选择左侧导航栏中开放 API->API 管理->API 列表：

点击 PCS API 行中开启按钮，完成授权访问目录名称的设置。以后所有个人云存储服务的文件全路径都会是以”/apps/授权目录/”开头。

3.4 第三方账号登录模块的配置

Frontia 的授权模块可以支持单点登录，但需要按照如下流程配置：

3.4.1 新浪微博单点登录

首先手机中必须安装新浪微博 3..0 以上版本才能够支持新浪微博单点登录。单点登录的流程如下（参考新浪微博开发者中心）：

1. 创建一个新浪微博开放平台的应用 A。
2. 创建一个百度开放平台的应用 B。
3. 将 A 的 appKey 和 appSecret 托管到 B，步骤参见：http://developer.baidu.com/wiki/index.php?title=docs/social/guide/nativeapp_login/nativeapp_login2。
4. 在 android 应用中显式打开对新浪单点登录的支持，注意要使用 A 的 appKey：mAuthorization.enableSinaWeiboSSO(SINA_APP_KEY_of_A);
5. 然后就可以使用了：

```
mAuthorization.authorize(yourActivity,
                FrontiaAuthorization.PlatformType.SINA_WEIBO, onSuccess, onFail);
```

3.4.2 QQ 单点登录

1. 创建一个腾讯互联平台的应用 A。
2. 创建一个百度开放平台应用 B。
3. 将 A 的 appKey 和 appSecret 托管到 B，步骤参见：http://developer.baidu.com/wiki/index.php?title=docs/social/guide/nativeapp_login/nativeapp_login2。
4. 在 android 应用中显式打开对新浪单点登录的支持，注意要使用 A 的 appKey：mAuthorization.enableQQSSO(QQ_APP_KEY_of_A);
5. 然后就可以使用了：

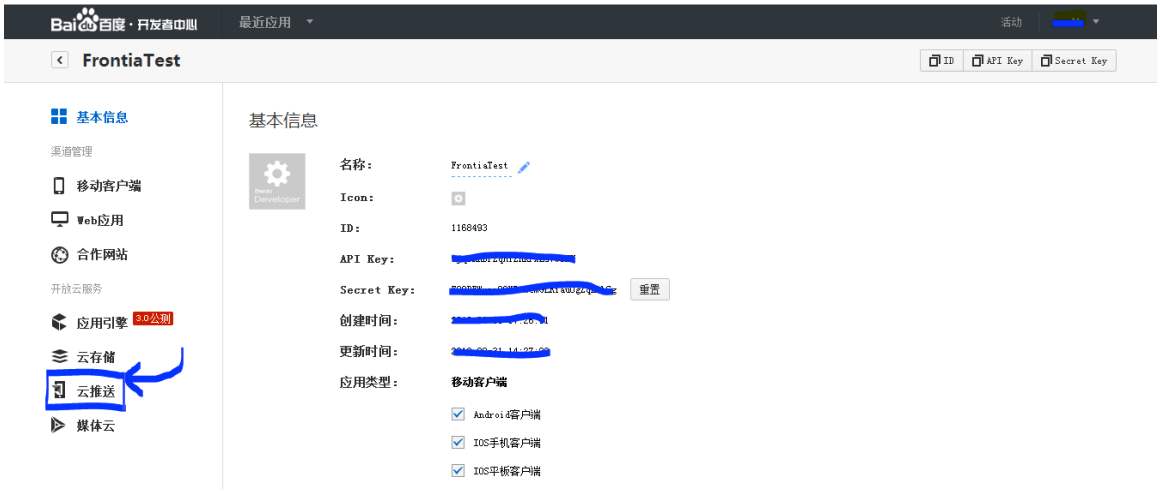
```
mAuthorization.authorize(yourActivity,
                    FrontiaAuthorization.PlatformType.QQ, onSuccess, onFail);
```

3.5 推送模块的配置

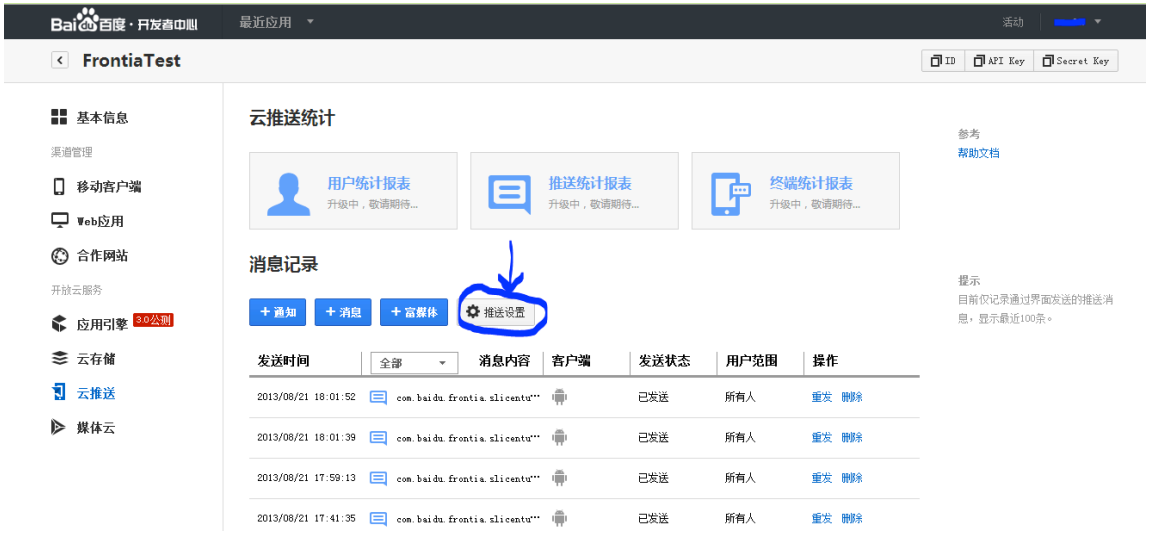
Frontia 的推送模块需要开发者在百度云推送的管理控制台注册自己的应用的包名。步骤如下：

1. 访问 http://developer.baidu.com/console ，点击应用名：

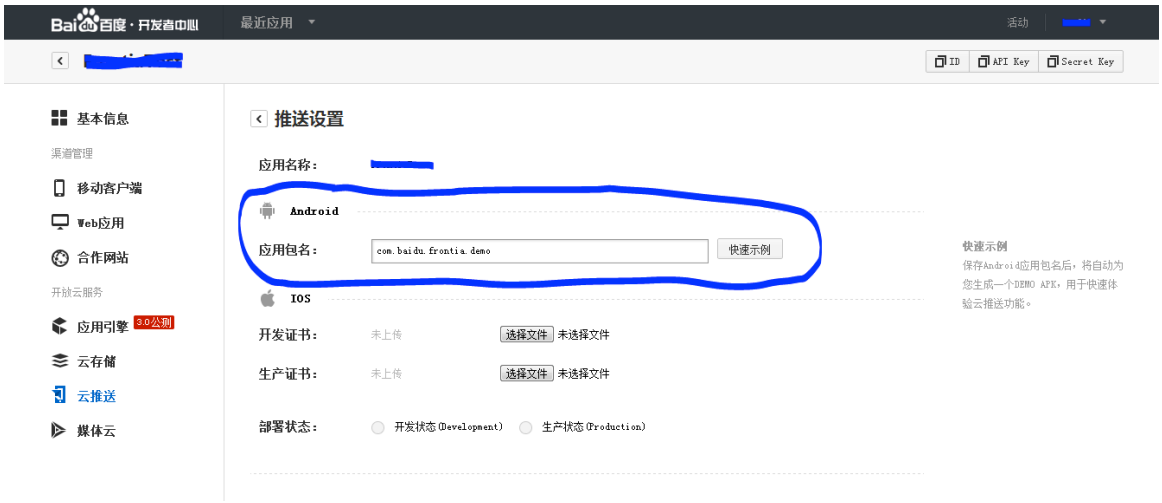
2. 在左侧“开放云服务下”点击“云推送”：



3. 点击“推送设置”：



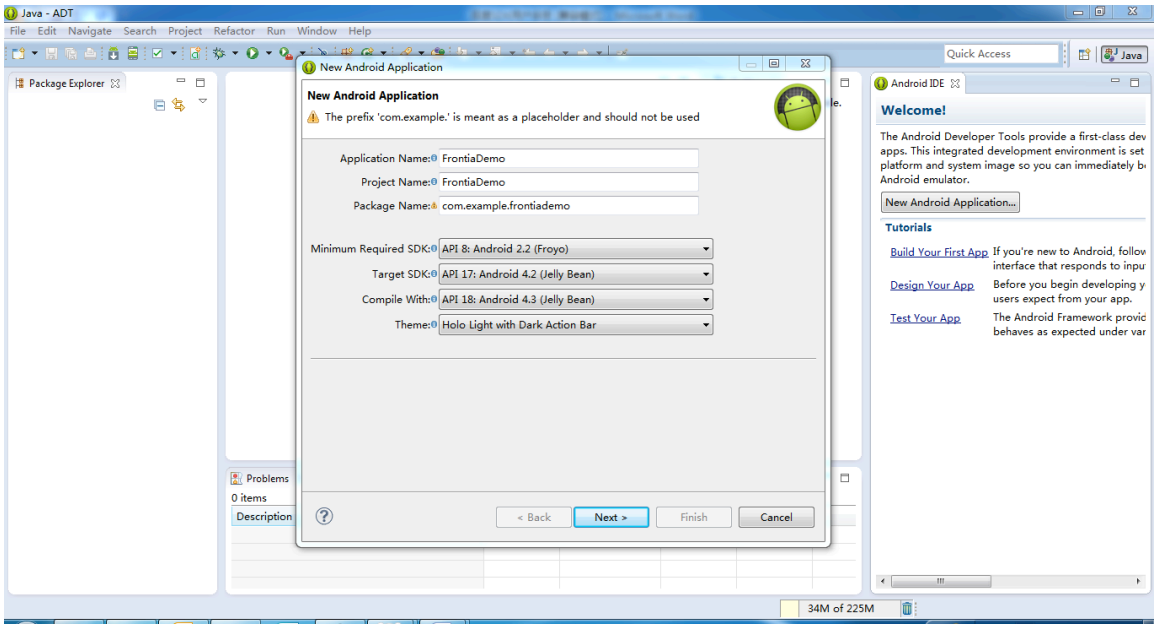
4. 在 Android 一栏里填写应用的包名：



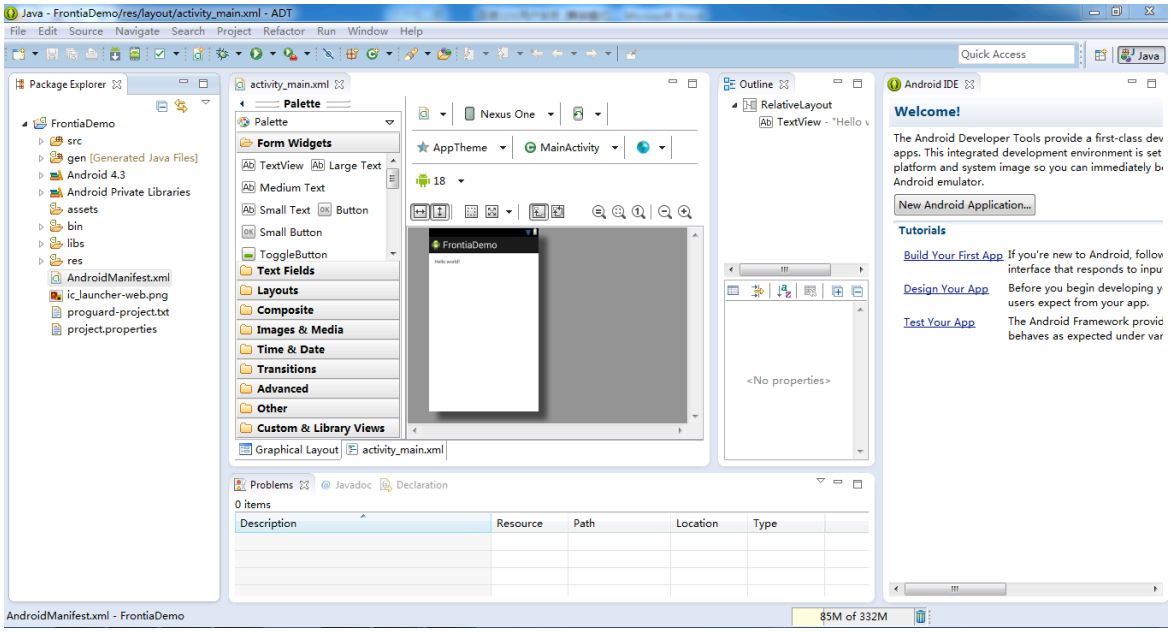
第4章　使用说明

4.1　新建工程

启动 eclipse，创建一个 Android 工程：

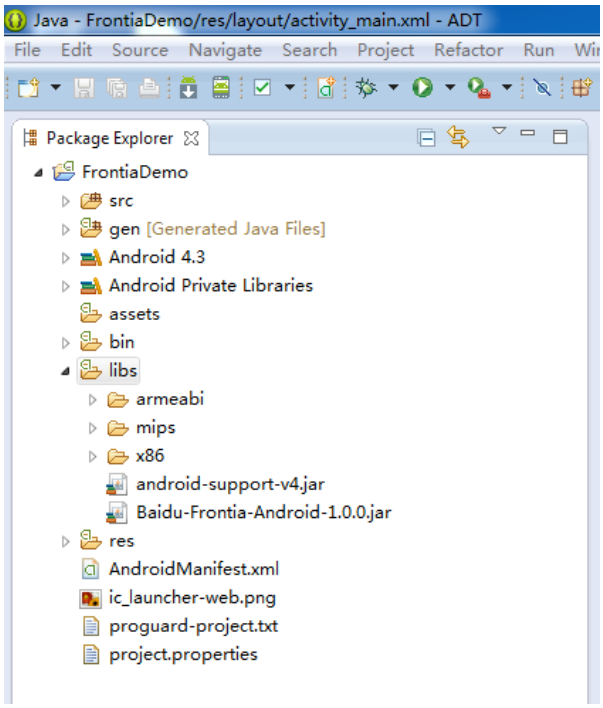


为方便起见，创建项目时全部使用默认设置。创建完后如下：



4.2 导入 Frontia

将解压后的 libs 目录下的所有内容复制到 Sample 工程的 libs 目录下：



4.3 配置 AndroidManifest.xml

双击工程目录里的 AndroidManifest.xml，添加如下权限：

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission android:name="android.permission.WRITE_SETTINGS" />
<uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW" />
<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
<uses-permission android:name="android.permission.BROADCAST_STICKY" />
<uses-permission android:name="android.permission.VIBRATE" />
<uses-permission android:name="android.permission.ACCESS_DOWNLOAD_MANAGER"/>
<uses-permission
android:name="android.permission.DOWNLOAD_WITHOUT_NOTIFICATION" />
<uses-permission android:name="android.permission.DISABLE_KEYGUARD" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
```

需要声明如下组件：

```
<receiver android:name="com.baidu.android.pushservice.PushServiceReceiver"
                android:process=":bdservice_v1">
  <intent-filter>
        <action android:name="android.intent.action.BOOT_COMPLETED" />
        <action android:name="android.net.conn.CONNECTIVITY_CHANGE" />
        <action
android:name="com.baidu.android.pushservice.action.notification.SHOW" />
        <action android:name="com.baidu.android.pushservice.action.media.CLICK" />
        <action android:name="com.baidu.android.pushservice.action.frontia.user" />
  </intent-filter>
</receiver>

<receiver android:name="com.baidu.android.pushservice.RegistrationReceiver"
                android:process=":bdservice_v1">
  <intent-filter>
        <action android:name="com.baidu.android.pushservice.action.METHOD" />
        <action android:name="com.baidu.android.pushservice.action.BIND_SYNC" />
  </intent-filter>
  <intent-filter>
        <action android:name="android.intent.action.PACKAGE_REMOVED"/>
        <data android:scheme="package" />
  </intent-filter>
</receiver>

<service android:name="com.baidu.android.pushservice.PushService"
          android:exported="true"
          android:process=":bdservice_v1" />

<receiver
       android:name="com.baidu.frontia.module.deeplink.db.DLDataPushReceiver">
  <intent-filter>
        <action android:name="com.baidu.android.pushservice.action.SDK_MESSAGE" />
        <action android:name="com.baidu.android.pushservice.action.sdk.RECEIVE" />
  </intent-filter>
  <intent-filter>
         <action android:name="android.intent.action.PACKAGE_REPLACED"/>
         <action android:name="android.intent.action.PACKAGE_ADDED" />
         <data android:scheme="package" />
  </intent-filter>
</receiver>
```

声明 FrontiaApplication：

1：如果项目中没有指定 application 名称，可以直接在<application>声明 FrontiaApplication：

```
<application android:name="com.baidu.frontia.FrontiaApplication"></application>
```

2：如果项目中已指定 application 名称，并且应用中使用的 application 直接从 android.app.Application 继承的，需要强制该 application 从 FrontiaApplication 继承:

```
<application android:name="XXX.XXX.MyApplication"></application>
```

```
public class MyApplication extends FrontiaApplication{
              @Override
              public void onCreate(){
                     super.onCreate();

              }
       }
```

3：如果项目中使用的 application 已经从第三方 application 继承，无法从 FrontiaApplication 继承的，

则可以在 application 的 onCreate（）方法里面调用 FrontiaApplication 的方法：

```
<application android:name="XXX.XXX.XApplication"></application>
```

```
public class XApplication extends ThirdApplication{
              @Override
              public void onCreate(){
                     super.onCreate();

                     FrontiaApplication.
                           initFrontiaApplication(getApplicationContext());
              }
       }
```

为了接收消息，需要继承 com.baidu.frontia.api.FrontiaPushMessageReceiver，并将其子类声明到 AndroidManifest.xml 中。假定 MyMessageReceiver 继承了该类，其声明如下：

```
<receiver android:name="com.baidu.frontia.demo.push.PushMessageReceiver">
   <intent-filter>
   <!-- 接收自定义消息 -->
   <action android:name="com.baidu.android.pushservice.action.MESSAGE" />
   <!-- 接收 bind,unbind,fetch,delete 等反馈消息 -->
   <action android:name="com.baidu.android.pushservice.action.RECEIVE" />
   <!— 接收通知消息的点击事件 -->
   <action
   android:name="com.baidu.android.pushservice.action.notification.CLICK" />
         </intent-filter>
      </receiver>
```

4.4 初始化 Frontia

在开始使用 Frontia 之前需要使用类方法传递入 3.2 申请的应用的 api key，完成初始化工作：

```
bool isInit = Frontia.init(getApplicationContext(), "your_app_key");
if(isInit){//Fronita is successfully initialized.
       //Use Frontia
}
```

第5章 API 说明

具体接口说明请参看开发包中的 docs 目录

第6章 示例

具体接口调用示例请参考开发包中的 demo 演示程序。

第7章 联系我们

如果以上信息无法帮助您解决在开发中遇到的具体问题，请通过以下方式联系我们：

邮箱: dev_support@baidu.com

百度 hi 官方技术讨论群：1411004

QQ 群：156179393

百度工程师会在第一时间回复您。

第8章 文档变更历史

版本	描述	变更日期
0.9	初稿	2013-08-27
1.0	修改稿	2013-8-31